

Easy Bluetooth (#30085)

The Easy Bluetooth is a RoboTech RBT-001 Bluetooth serial module with an adapter specifically designed to be used with the Parallax Board of Education® AppMod Header or breadboard. In addition, the Easy Bluetooth can plug into any 0.1 inch spacing development platform which allows it to be breadboard friendly, and also easy to implement into a soldering application. The module has two parts, the RBT-001 module and the SIP with voltage regulator PCB. With the on-board regulator, the module can be connected to voltages higher than 3.3 VDC, such as the Board of Education regulated supply (+5 VDC) without worry of damaging the unit; while the RX and TX can utilize serial communication at CMOS and TTL levels.

Features

- 1.x & 2.0 Bluetooth Compliant
- Class 2 Operation (nominal range up to 30 meters)
- 10-pin SIP package for breadboard, perfboard, or Board of Education AppMod Header
- On-board regulator for safe operations across various voltages
- CMOS & TTL Compatible

Key Specifications

- Power requirements: 3.3 to 5.5 VDC
- Communication: UART Command/Data Port supports for up to 921.6k baud
- Operating temperature: +32 to +113 °F (0 to +45 °C)
- Dimensions: 1.40 x 1.79 x .49 in (34.41 x 45.65 x 12.51 mm)

Application Ideas

- Control a Boe-Bot via Bluetooth from a PC, Cell Phone, or another Bluetooth module
- Communicate with a device or project wirelessly

Packing List

- RBT-001 Module
- 10-pin SIP with connector Module (pre-assembled with RTB-001)

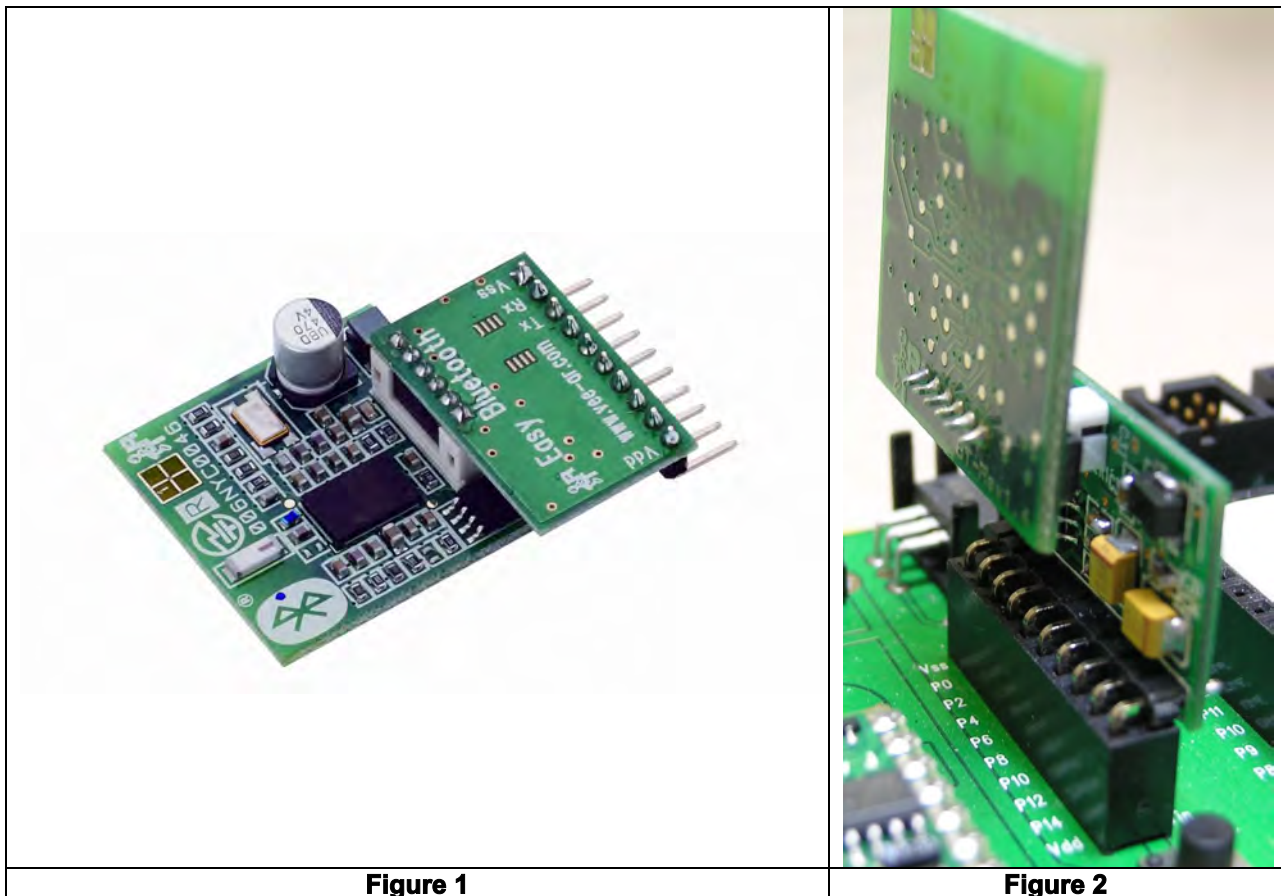


Quick Start Circuit for the BASIC Stamp 2 and Board of Education

There are a few steps to take to install an Easy Bluetooth module, creating a Bluetooth connection on a PC operating with Windows XP, and finally writing a program for the BASIC Stamp[®] 2. Once you are done, you will have a working Easy Bluetooth module communicating with the PC via Bluetooth.

Installing the Easy Bluetooth Module

1. Carefully open the Easy Bluetooth package and check that the two small boards are properly plugged into each other as shown in Figure 1.
2. Now plug the Easy Bluetooth module into the AppMod header of the Board of Education: insert the module in the left row of the AppMod Header and notice there are labels on the module that indicate correct pin placement (Rx uses P0, Tx uses P2, Vss with Vss and Vdd with Vdd). Visually confirm the module is inserted correctly using Figure 2 before powering.
3. Connect a communication cable and power supply to the Board of Education and turn the power switch to location 1.



Creating a new Bluetooth Connection

1. Be sure that the Bluetooth dongle that you installed on your PC is working correctly and is turned on (see Bluetooth Dongle user manual for specific instructions to ensure proper operation).
 - a. Open Control Panel, click "Printer and Other Hardware" and then click Bluetooth Devices. If in Classic View double click "Bluetooth Devices".

- b. Click "Add".
 - c. Check "My Device is set up and ready to be found" in the "Add Bluetooth Device Wizard" and click "Next".
 - d. Select "EasyBT" from the displayed Bluetooth devices and click "Next".
 - e. Select "Use the passkey found in the documentation" and enter the Passkey Code 0000 (zero four times) and click "Next".
 - f. Bluetooth Manager should display and designate an Outgoing and Incoming COM port for the Easy Bluetooth device: make a note of the COM ports because these are the ports you will use to communicate with the Easy Bluetooth.
 - g. Select "Finish" to complete the Bluetooth device configuration.
2. To close the "Bluetooth Devices" window click "OK".

A test for the BASIC Stamp 2 microcontroller

1. The Easy Bluetooth can communicate at numerous baud rates, and with many different microcontrollers. This example uses the AppMod Header on the Board of Education, the BASIC Stamp 2, at 9600 baud.
2. The easiest way to see in your code if a Bluetooth connection has been established is to wait for a byte to be received. You can copy the program below into the BASIC Stamp Editor to ensure a proper connection.

```
' {$STAMP BS2}
' {$PBASIC 2.5}

RX          CON    2      'Receive Pin
TX          CON    0      'Transmit Pin
Baud        CON    84     '9600 Baud

combyte     VAR Byte  'Communication Byte

DEBUG "Use This Screen for Display",CR

DO
  'Wait for a first byte indicating an active Bluetooth connection
  SERIN RX, Baud, [combyte]
  DEBUG combyte
LOOP
```

3. Download the program using Run => Run (ctrl + r) from the BASIC Stamp Editor
4. Keep Current DEBUG Screen open, and open another DEBUG Screen (ctrl +d) from the BASIC Stamp Editor, and select TX COM that the Bluetooth module (refer to COM ports during Bluetooth installation) is using; you should now have 2 DEBUG Terminal windows open. Click in the white area of the DEBUG window of DEBUG Terminal 2 and press any key; if all the steps were correctly done, the same key will be sent back to the DEBUG Terminal 1 window.

Resources and Downloads

You may download the RBT-001 user manual, example source codes, PC applications, and mobile phone application from www.parallax.com.

*Mobile phone must be compatible with the JSR-82 Java API; usually all Nokia and Sony Ericsson phone are compatible, but check users manual for compatibility. Configuration of mobile phone is not supported by Parallax Technical Support.

Device Information

Theory of Operation

Bluetooth is an open wireless protocol for exchanging data over short distances from fixed and mobile devices, creating Personal Area Networks (PANs). It was originally conceived as a wireless alternative to RS232 data cables. It can connect several devices, overcoming problems of synchronization.

There are various versions of Bluetooth communication and the Easy Bluetooth is compatible with versions 1.x and 2.0.

Precautions

The Easy Bluetooth uses the microwave radio frequency spectrum in the 2.4 GHz to 2.4835 GHz range. Maximum power output from this Bluetooth radio 2.5 mW; Class 2 devices are considered less of a potential hazard than mobile phones.

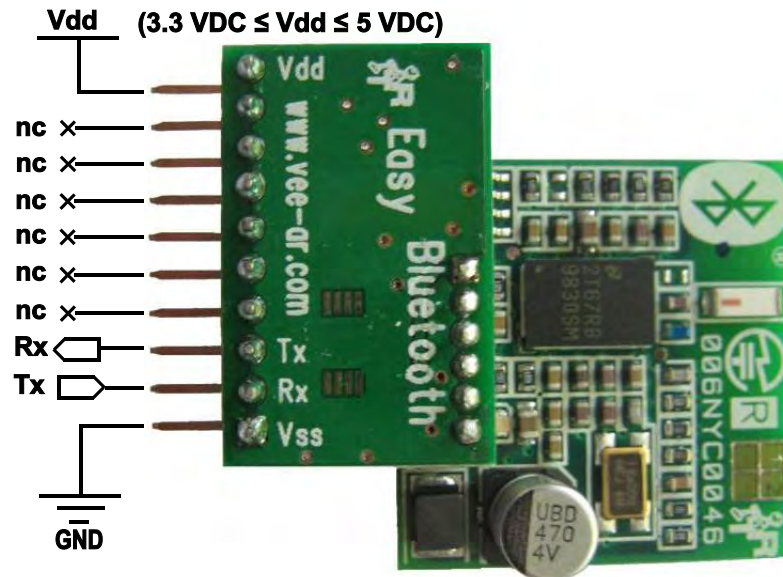
Pin Definitions and Ratings

Pin	Name	Function
1	Vss	Ground
2	RX	Receive Serial Communication (CMOS & TTL Level Compatible)
3	TX	Transmit Serial Communication (CMOS & TTL Level Compatible)
4	NC	Not Connected
5	NC	Not Connected
6	NC	Not Connected
7	NC	Not Connected
8	NC	Not Connected
9	NC	Not Connected
10	Vdd	+3.3 to +5 VDC (+5.5 max voltage)

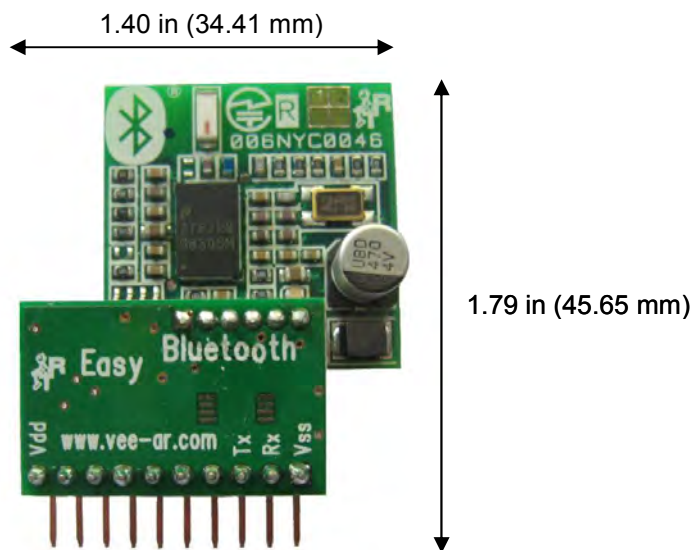
Communication Protocol

UART Command/Data Port supports up to 921.6k non-inverted baud.

Connection Diagram



Module Dimensions



Troubleshooting

From time to time there may be some snags that can cause what would seem like malfunctions in the module. If you experience any of the symptoms listed below, here are some quick fixes to try.

Q1. Will not connect to Easy Bluetooth module?

A1. Verify that the Bluetooth device has been properly installed via the Device Manager; there may be drivers needed depending on the type of Bluetooth dongle/module you are using.

Q2. Bluetooth port isn't available for selection in the new DEBUG window.

A2. Be sure that the com port is included in the port search for the BASIC Stamp editor; for details on how to do this, please refer to the BASIC Stamp Help file in the editor (version 2.5 or later)

Q3. Will not power up?

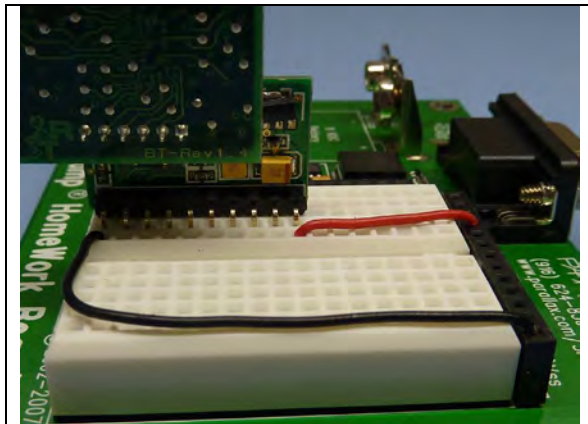
A3. Verify that the module is connected correctly and that Vdd and Vss have not been swapped.

Q4. Receiving Easy Bluetooth connection LED (green) blinks, but no information is displayed on the DEBUG screen?

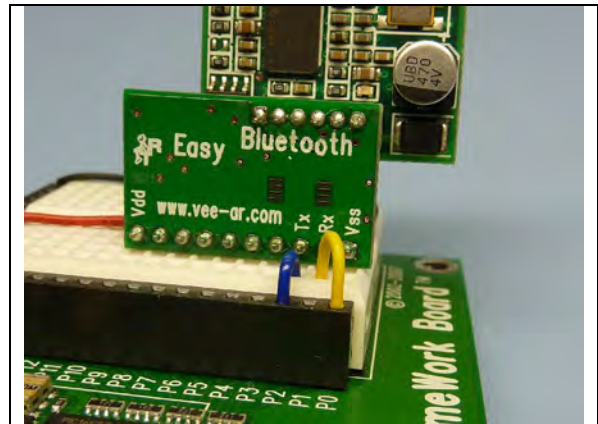
A4. If the receiving modules LED is blinking, then there is a completed connection and the setting in the debug window is not set properly. Verify that the correct com port and baud rate is selected in the terminal window. If com port does not come up, please see A2 above.

Q5: What if I don't have a Board of Education with an AppMod Header; can the Easy Bluetooth be used with the HomeWork Board or a breadboard?

A5: Yes you may. There are 4 connections needed to correctly connect the Easy Bluetooth module to a microcontroller, the Rx, Tx, Vss (ground) and Vdd (+5VDC). Here is an example that connects the Easy Bluetooth module to the HomeWork Board, but could as easily be another type of controller with the provided connections. Here are two pictures with the connections made to a HomeWork Board. Please keep in mind, the connections match the example program provided.



Easy Bluetooth module front connection



Easy Bluetooth module back connection

Example Application

This example application uses the RoboTech SRL GUI interface to control a Boe-Bot[®] robot via Bluetooth.

*A note to mention is that this application uses a standard operational Boe-Bot; so please have an operational Boe-Bot ready while completing this example. For help building a fully operational Boe-Bot, please refer to Robotics with the Boe-Bot Text.

To install the GUI software, complete the following steps:

1. Install the Easy Bluetooth PC software from the "Easy Bluetooth Software.zip" file with all the default and install locations as prompted by the installer. At the end of the installations you will find a new program folder "Easy Bluetooth" under the "Parallax Inc" main folder named with a sub folder named "BoeBot Remote Control", in your Start Menu.
2. Select "BoeBot Remote Control" application from the sub-folder listed in the previous step.
3. Click "Options" and select the COM port that was previously established with the Bluetooth configuration.
4. Download the program below using the BASIC Stamp Editor (ctrl + r) to the BASIC Stamp 2 and then download the program below to the BASIC Stamp 2 using Run => Run or (ctrl + r).
5. Disconnect the Boe-Bot from PC
6. Click "Connect" in the BoeBot Remote Control application to gain control of the Boe-Bot.

BASIC Stamp® 2 Program

```
' =====
'
' File..... Easy Bluetooth.bs2
' Purpose... Control a Boe-Bot with a GUI interface
' Author.... Technical Support & RoboTech SRL
' E-mail.... support@parallax.com
' Started... April 1st 2009
' Updated... N/A
'
' {$STAMP BS2}
' {$PBASIC 2.5}
'
' =====

' -----[ Program Description ]-----

' This program is what is loaded into the BASIC Stamp prior to using the GUI interface
' from RoboTech SRL. It allows a Boe-Bot to be controlled via a GUI interface.
'
' Controls are as followed:
'
'   Up Arrow    = Forward
'   Down Arrow  = Backwards
'   Left Arrow  = Left Turn
'   Right Arrow = Right Turn
'   Space Bar   = Stop
'
' You can press and hold and release an arrow key for 2 seconds to access a double
' speed of each action. For example, Press Up Arrow, release to make the Boe-Bot move
' faster forward.
```

```

' -----[ I/O Definitions ]-----
BT_RX          PIN      2          ' RX of the Easy Bluetooth
BT_TX          PIN      0          ' TX of the Easy Bluetooth
LED            PIN      5          ' Indicator LED for Bluetooth Connection

' -----[ Constants ]-----
BT_SPEED        CON      84          ' baud 9600 true UART

' -----[ Variables ]-----
tLeft          VAR      Word        ' Left Servo control pulse durations
tRight         VAR      Word        ' Right Servo control pulse durations
temp           VAR      Word        ' Temp variable

' Buffer array not declared as buffer VAR Word(5) for SERIN functionality.
' It can still be accessed as buffer(0), buffer(1), etc. However,
' buffer0, buffer1, etc. should be used in SERIN commands with variations
' of WAIT.

buffer0        VAR      Byte        ' Buffer - Start char = $ff
buffer1        VAR      Byte        ' Message Index value
buffer2        VAR      Byte        ' Command
buffer3        VAR      Byte        ' Argument 1 (return data 1)
buffer4        VAR      Byte        ' Argument 2 (return data 2)
buffer         VAR      buffer0     ' For standard array indexing
msgIndex       VAR      Byte        ' message index
rxrc           VAR      Byte        ' Receive Clear

' -----[ Initialization ]-----
Program Start:
  LOW LED
  DEBUG CLS

  PAUSE 1000          ' Wait for the RBT-001 radio to be ready.

  msgIndex = 0

  buffer0 = $FF          ' Connection packet
  buffer1 = msgIndex
  buffer2 = $CC
  buffer3 = 100
  buffer4 = 100
  GOSUB Set_Servo_Speed

  DEBUG CR,"Waiting connection..." ' wait for connection request
  SERIN BT_RX, BT_SPEED, [WAITSTR buffer \ 3, buffer3, buffer4]

' -----[ Program Code ]-----
DO
  SELECT buffer2
  CASE $CC              ' Connect
    HIGH LED
    msgIndex = 0
    DEBUG CR,"Connected"
    GOSUB Reply

```

```

CASE $DD                                ' Disconnect
  LOW LED
  DEBUG CR,"Disconnected"
  GOSUB Reply
  GOTO Program_Start
CASE $11                                ' Servo
  DEBUG CR,"Servo"
  GOSUB Set_Servo_Speed
  GOSUB Reply
CASE ELSE                                ' Error
  buffer2 = $EE
  GOSUB Reply
ENDSELECT

Resume:                                  ' If Message not rcvd, try again

PULSOUT 13, tLeft                        ' Servo control pulses
PULSOUT 12, tRight

SERIN BT_RX, BT_SPEED, 10, Resume, ' Get next command
  [WAITSTR buffer \ 2, buffer2,
   buffer3, buffer4]

PULSOUT 13, tLeft                        ' Servo control pulses again
PULSOUT 12, tRight
LOOP

Reply:
  msgIndex = msgIndex + 1                ' Increment message index for reply.
  buffer1 = msgIndex                     ' Next message from PC has to use reply's buf[1].
  SEROUT BT_TX, BT_SPEED, [STR buffer \5]

' -----[ Subroutines ]-----

Set_Servo_Speed:                         ' Maps to 650 to 850 with 750 stopped.
  tLeft = buffer3 + 650                  ' Decode servo speed.
  tRight = buffer4 + 650
RETURN

' -----[ Data ]-----

ResetOnOff          DATA 0              ' On/off toggle w/ Reset button
RequestConnect       DATA $FF, 0, 1, 0, 0
ConnectionGranted    DATA $FF, 0, 2, 0, 0
RequestCommand       DATA $FF, 0, 3, 0, 0
ServoSpeeds          DATA $FF, 0, 4, 0, 0

```

RoboTech Bluetooth Serial Module RBT-001



User Manual Ver. 1.2

RoboTech srl

Via Mazzini 82
19038 Sarzana (SP) – ITALY
P.IVA (VAT Number): IT 01185460118

Sede Operativa (Main Office):
Via Boccioni 1
56037 Peccioli (PI) – ITALY
Tel: +39.0587.672027
Fax: +39.0587.670936

E-mail: info@RoboTechsrl.com
<http://www.RoboTechsrl.com>

Copyright © 2004-2006 RoboTech srl
All rights reserved.

RoboTech srl assumes no responsibility for any errors, which may appear in this manual. Furthermore, RoboTech srl reserves the right to alter the hardware, software, and/or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. RoboTech srl products are not authorized for use as critical components in life support devices or systems.

The Bluetooth trademark is owned by the Bluetooth SIG Inc., USA, and is licensed to RoboTech srl.

All other trademarks listed herein are owned by their respective owners.

Table of Contents

Product Description	6
Technical specifications	7
Physical dimensions and pin assignment	7
Recommended Operating Conditions	8
Digital DC Characteristics	8
Power Supply Requirements ¹	8
Powering with Voltage DC Input above 3.0V	9
RS232 adapter board schematic	10
UART Interface	11
Command mode and Transparent mode	12
Command Mode	12
Transparent Mode	12
Leaving transparent mode	12
Bluetooth mode	13
NVS Settings influencing the operation state	13
Automatic operation on/off	15
Force Master on/off	16
Default connections stored in NVS	16
Default connection transparent On/Off	16
Operation States	16
Idle, Automatic and Idle, Non-automatic	18
Incoming Link at Idle Automatic	18
Incoming Link in Idle Non-Automatic	18
Piconet Master	18
Scatternet Master	19
Transparent Master	19
Single Slave	20
Scatternet Slave	20
Transparent Slave	20
UART Protocol Principles	22
Framing	22
Start delimiter	22
Packet type identification	22
Opcode	22
Data length	25
Packet data	25
Checksum	25
End delimiter	25
Retransmission	25
Flow control	25
Byte Order	25
Commands in <i>Command Mode</i>	26
Searching for remote devices	26
Inquiry	26
Device Found	26
Getting user-friendly name of remote Bluetooth Device	27
Remote Device Name	27

Reading the user-friendly name of the local device.....	28
Read Local Name	28
Changing the user-friendly name of the local device.....	28
Write Local Name	28
Reading the Bluetooth device address of the local device	29
Read Local Bluetooth Address	29
Changing Scanmode (Connectability and Discoverability).....	29
Set Scan Mode	30
Set Scan Mode Indication	30
Bluetooth Security	31
Get Fixed PIN	31
Set Fixed PIN	31
PIN request.....	32
Get Security Mode	32
Set Security Mode	33
Pairing	33
Remove Pairing	33
List Paired Devices	34
"Air interface" Low Power Modes	34
Exit Sniff Mode	35
Enter Park Mode	35
Exit Park Mode	36
Enter Hold Mode	36
Set Link Policy	37
Get Link Policy.....	37
Power Save Mode Changed	38
Device Low Power Modes.....	38
Disable Transport Layer	38
ACL indications	39
ACL Established	39
ACL Terminated.....	39
Serial port profile (SPP).....	40
Open / Close an RFCOMM port	40
Reading / changing current port configuration.....	40
Set Port Config	40
Get Port Config	41
Port Config Changed	42
Reading / changing current port status.....	43
Get Port Status	43
Set DTR.....	44
Set RTS	45
Set BREAK	45
Set Overrun Error	46
Set Parity Error	46
Set Framing Error	46
Port Status Changed	47
Establish / Release SPP link to a remote device.....	48
Establish Link	48
Link Established.....	49
Incoming Link Established	49

Release Link.....	49
Link Released.....	50
Sending / Receiving data on a SPP link (Normal mode)	50
Send Data.....	50
Incoming Data	51
Sending / Receiving data on a SPP link (Transparent mode)	51
Transparent Mode	52
Default Connection.....	53
Connect Default Connection.....	53
Store Default Connection.....	53
Get List of Default Connections	54
Delete Default Connection.....	54
Link supervision Timeout.....	55
Set Link Timeout.....	55
Get Link Timeout	55
Service discovery application profile (SDAP)	56
SDAP Connection Handling	56
Connect	56
Disconnect.....	57
Connection Lost.....	57
SDAP Service Discovery	57
Service Browse.....	57
Service Search	58
Service Request	59
Attribute Request.....	59
Configuration	61
Change NVS UART Speed	61
Change UART Settings	61
Read Operation Mode.....	62
Write Operation Mode	62
Set Ports To Open.....	62
Get Ports To Open	63
Restore Factory Settings.....	63
Store Class of Device.....	63
Force Master Role.....	64
Set Default Link Policy	64
Get Default Link Policy	65
Set Event Filter.....	65
Get Event Filter	66
Set Default Link Timeout.....	66
Get Default Link Timeout.....	66
Set Default Link Latency	67
Get Default Link Latency	67
SDP Record Handling	67
Enable SDP Record.....	68
Delete All SDP Records.....	68
Store Generic SDP Record.....	69
Other Commands.....	69
Reset.....	69
Device Ready.....	70

Test Mode	70
RF_TEST_MODE.....	70
Read RSSI	71
Read NVS	72
Write NVS	72
ErrorCodes	73
ACL Error Codes.....	73
Generic error codes.....	74
RFCOMM Error Codes.....	76
RFCOMM Release Reasons.....	76
Bibliography	77
Certifications	78
Bluetooth	78
CE	78
Revision History	79

Product Description

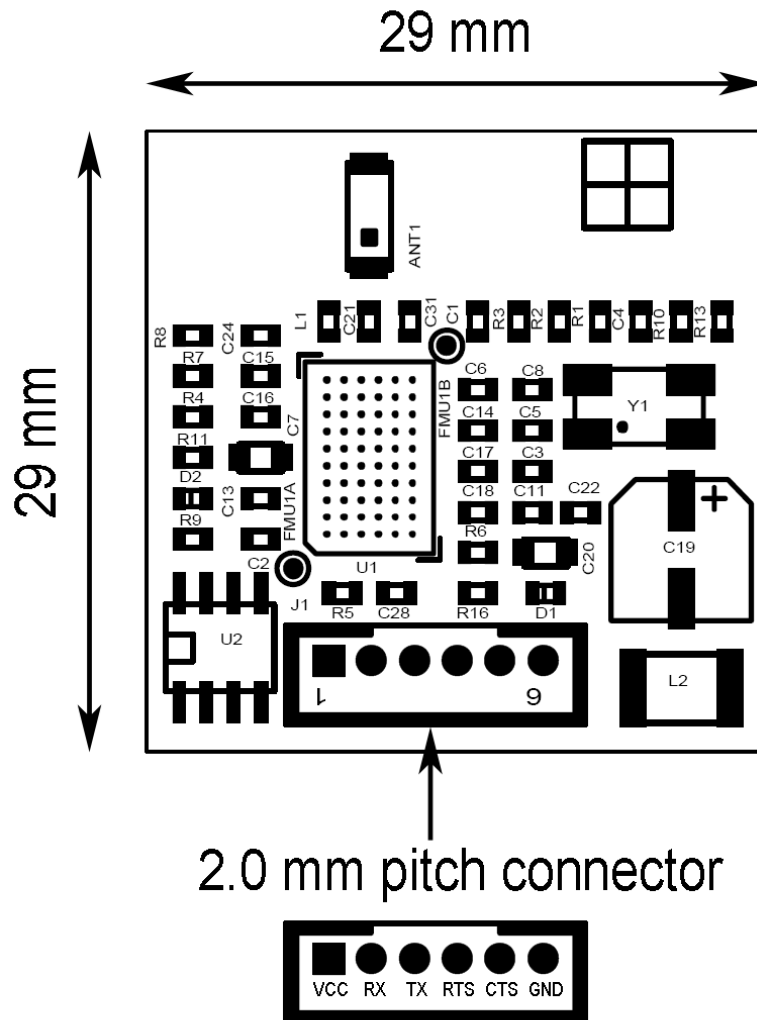
The RoboTech Bluetooth Serial Module is an effective and low-cost solution to free your hardware applications from wires.

Main features are:

- Compliant with the **Bluetooth 2.0** Specification
- Certified as an **end product**: no additional Bluetooth qualification is needed when using this module
- Backwards compatible to Bluetooth 1.x versions
- Class 2 operation (nominal range up to 30m)
- Low power consumption
- UART Command/Data Port supports for up to 921.6k baud rate
- Profiles: GAP, SDAP, SPP
- Integrated chip antenna
- Support for Adaptive Frequency Hopping (AFH) and 802.11 co-existence
- Small size (29x29mm)
- RoHS compliant
- Radio Type Approved for Europe and Japan

Technical specifications

Physical dimensions and pin assignment



Pin number	Pin name	Type (Input/Output)	Description
1	VCC	I	Voltage DC input (Typical 3VDC)
2	RX	I	Serial Port Receive Data (TTL level)
3	TX	O	Serial Port Transmit Data (TTL level)
4	RTS	O	Serial Port Request To Send (active low)
5	CTS	I	Serial Port Clear To Send (active low)
6	GND	-	Ground

Recommended Operating Conditions

Symbol	Parameter	Min	Typ	Max	Unit
VCC	Voltage DC Input	2.5	3.0	3.3 ¹	V
Ta	Ambient Operating Temperature Range	0	25	45	°C

1. Recommended maximum 3.0V for best RF performance

Digital DC Characteristics

Symbol	Parameter	Condition	Min	Max	Unit
V _{IH}	Logical 1 Input Voltage High	2.5V ≤ VCC ≤ 3.0V 3.0V ≤ VCC ≤ 3.3V	0.7 x VCC 2.0	VCC + 0.2 VCC + 0.2	V
V _{IL}	Logical 0 Input Voltage Low	2.5V ≤ VCC ≤ 3.0V 3.0V ≤ VCC ≤ 3.3V	-0.2 -0.2	0.25 x VCC 0.8	V
I _{OH}	Logical 1 Output Current	V _{OH} = 2.4V, VCC = 3.0V	-10		mA
I _{OL}	Logical 0 Output Current	V _{OL} = 0.4V, VCC = 3.0V	10		mA

Power Supply Requirements¹

Symbol	Parameter	Min	Typ ²	Max	Unit
I _{RXSL}	Receive Data in SPP Link, Slave ³		26		mA
I _{RXM}	Receive Data in SPP Link, Master ³		23		mA
I _{SnM}	Sniff Mode, Sniff interval 1 second ³		5.6		mA
I _{SC-TLDIS}	Scanning, No Active Link, TL Disabled ³		0.43		mA
I _{Idle}	Idle, Scanning Disabled, TL Disabled ³		100		µA

1. Based on UART Baudrate 921.6kbit/s.

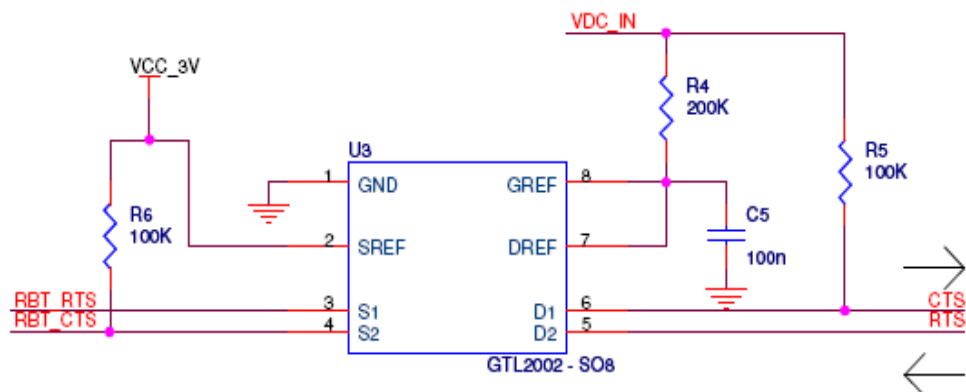
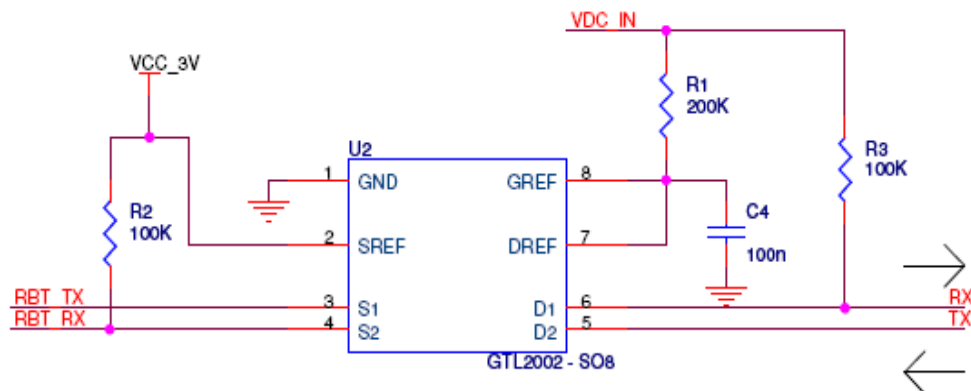
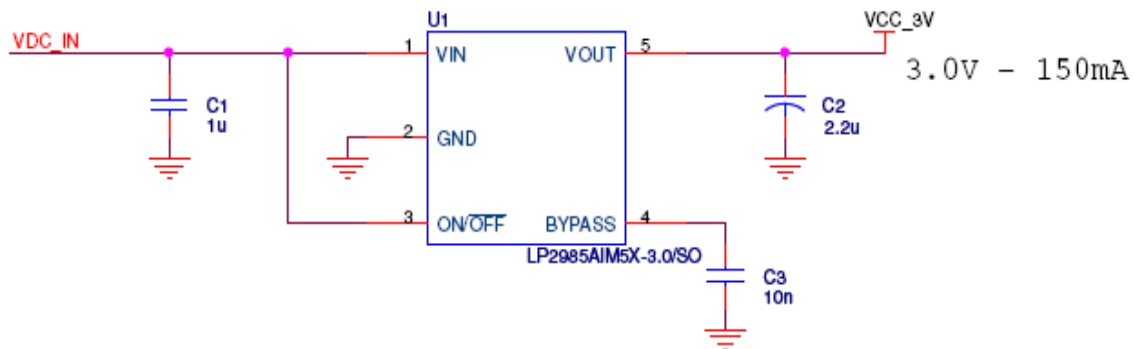
2. VCC = 3.3V, Ambient Temperature = +25 °C.

3. Average values excluding Leds

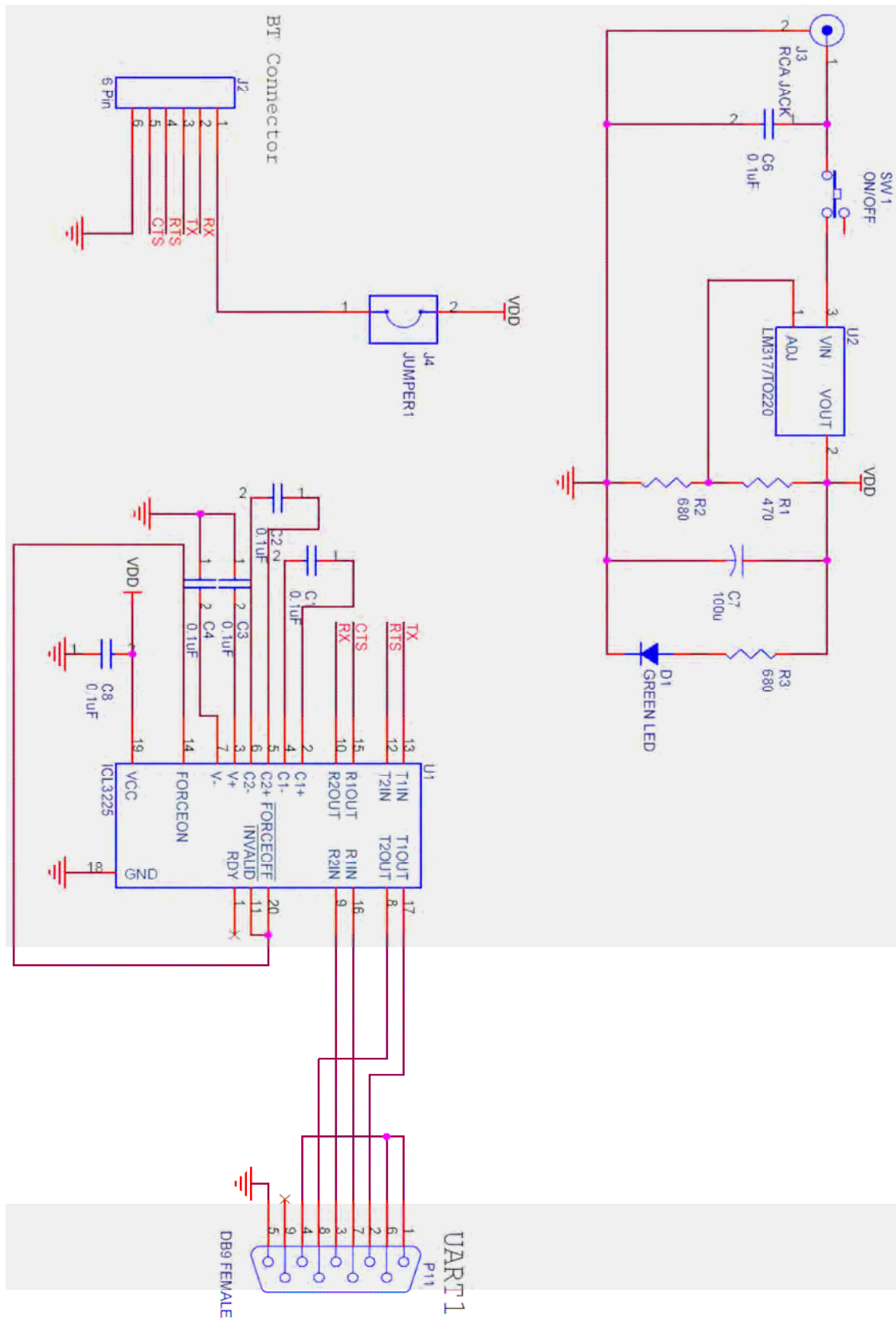
TL= Transport Layer

Powering with Voltage DC Input above 3.0V

If the Host circuitry works with a voltage above 3.0V, a regulator is needed to provide the correct Voltage DC Input level to the RBT-001 module keeping the best RF performance. Furthermore the RBT-001 module's inputs must be lowered to acceptable levels. Just for instance, a simple schematic considering a Host circuitry working from 3.3VDC to 5VDC is reported here below.



RS232 adapter board schematic



UART Interface

The main communication interface between the RBT-001 and the host is the UART Interface. The UART interface between host and RBT-001 needs to be connected in Null Modem configuration, meaning RTS/CTS and TX/RX are crossed.

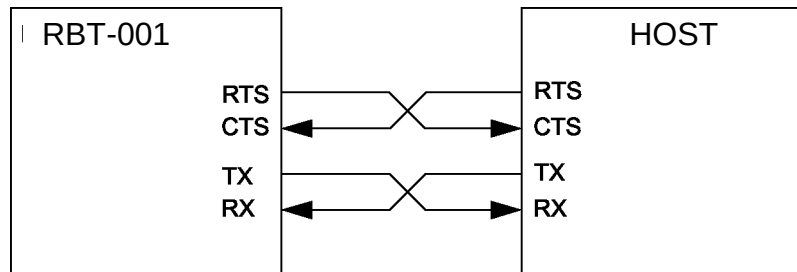


Figure 1: UART Null Modem connection

The command interface is based on a command/event based structure. Each command (also called “Request”) will be acknowledged with the appropriate status event (also called “Confirm”). Unexpected events, like incoming link establishment or data, are also sent as events, but signed with a different package type called “Indicator”. Please see the section “Commands in *Command Mode*” on page 6 for the detailed description of the command interface.

Since the RBT-001 acts like a gateway between the Bluetooth Link and the UART interface, the UART connection should use 4-wire hardware handshaking for optimal buffer handling. The RBT-001 uses the RTS lines to indicate low buffers and reacts on the CTS from the host, immediately stopping sending packages to the host.

In case the host microcontroller is not able to provide hardware handshaking, the UART should be used in “Command Mode” only, since the device would not be able to indicate full buffers by using the RTS signal. In the same way, the host needs to provide enough buffer space to be able to handle the incoming data, since it’s not able to stop the flow from the RBT-001.

Command mode and Transparent mode

Command Mode

The RBT-001 offers a wide range of commands to configure the hardware and the Bluetooth operation. As the command set is on top of the profiles, Bluetooth operational commands are reduced to high level commands controlling general Bluetooth operation.

In Command Mode, the RBT-001 will try to interpret all data sent over the UART to a known command. The commands have to be sent in a specific package format. The interface is based on an event mechanism. Any command sent will be confirmed by the appropriate confirmation event. Unexpected events (e.g. incoming links) will be reported by indication events.

Please refer to the following section for a complete list of commands and their usage.

Transparent Mode

In case the RBT-001 has established a link to only one remote device and no configuration commands have to be sent to the RBT-001 ("Command Mode"), the UART interface can be switched to "Transparent mode". This means data are directly routed to the Bluetooth link and not interpreted. Also incoming data are not indicated as events, they are sent as RAW data to the UART.

Leaving transparent mode

As the RBT-001 does not listen to commands, UART Break has to be used to tell the device to leave the transparent mode.

Bluetooth mode

The operation of the RBT-001 can be divided into different states. Each state represents special situations and describes the behaviour of the module.

The operation states are influenced by a few features, stored in the NVS (*Non-Volatile Storage*). Depending on those, the device will come up to different modes and act according to those settings:

NVS Settings influencing the operation state

Before actually describing the different states, we should have a look on the parameters, which influence the behaviour of the RBT-001. The following parameters are stored in NVS and are checked during boot-up.

RBT-001 System Parameters, NVS EEPROM Memory Map

No.	Address	Parameter	Default Value	Description	SW Reset required
1	0000-0005	BD_Addr	<empty>	Bluetooth Device Address LAP(Isb), LAP, LAP, UAP, NAP, NAP (msb) Required for Bluetooth mode	no
2	0006	-	0x00	Reserved	-
3	0007	Unit-KeyPresent	0xFF	Used by BT core, generated during pairing procedure.	no
4	0008-0017	UnitKey	0xFF..0xFF	Used by BT core, generated during pairing procedure.	no
5	0018	Device-Name-Length	0xFF	Length of Parameter 6 "Devicename"	no
6	0019-0040	Device-Name	0xFF...0xFF	Friendly Name of the Bluetooth Device	no
7	0041	Country-Code	0x00	Used by BT core	yes
8	0042	PinLength	0x04	The length of parameter 9, "PinCode". In case set to 0, the RBT-001 will request pin from host.	no
9	0043-0052	PinCode	"0000"	Fixed PinCode used for pairing with other devices	no
10	0053-0055	ClassOfDevice	0x000000	The 'Class of Device' describes general functionality of the Bluetooth Device and is transmitted during the Inquiry process.	no
11	0056	SppPorts-ToOpen	0x00000001	Bitmask defining the RFCOMM channels to open. For each channel one RFCOMM instance will be created.	no
12	005A	Preferred-MasterRole	0x00	Preferred Master forces the device to switch to Master Role after being connected. The device will reject the link if command could not be executed.	yes

No.	Address	Parameter	Default Value	Description	SW Reset required
13	005B	Automatic Operation	0x01	Configures the general behavior of the device. 0x00: Automatic OFF (Non-automatic) 0x01: Automatic ON (Automatic)	yes
14	005C	PageScan-Mode	0x01	Configures the connectability of the device 0x00: not connectable 0x01: normal scan 0x81: interlaced scan (faster connection time)	no
15	005D	Inquiry-ScanMode	0x01	Configures the discoverability of the device 0x00: not discoverable 0x01: normal scan 0x81: interlaced scan (faster response time)	no
16	005E	Security-Mode	0x02	Configures Service Level Security Mode.	no
17	005F-0060	Default-LinkPolicy	0x000F	Configures the default link policy for incoming links.	no
18	0061	EventFilter	0x01	Configures the level of events reported to the host. 0x00: No filter, all events reported 0x01: ACL events filtered, only API events reported. 0x02: All events filtered, only UART breaks indicated 0x03: All events filtered, including UART break.	no
19	0062	-	0xFF	Reserved	-
20	0063-0064	LinkTime-out	0x7D00	Configures the default link supervision timeout (in slots, 0.625ms) used for incoming and outgoing links.	no
21	0065	-	0x00	Reserved	-
22	0066	-	0x00	Reserved	-
23	0067-0068	RfcommLatency	0x0000	Configures the default poll period of master to slave. 0x0000: No requirement (default 40slots) 0x0002-0x0190: Valid link latency	no
24	0069-006C	-	0x00000000	Reserved	-
25	006d	UartParity-Bit	0x00	Parity setting for the hardware UART interface. 0x00: No Parity 0x01: Even Parity 0x02: Odd Parity	yes
26	006E	UartStopBit	0x00	Stop bit settings for the hardware UART interface 0x00: 1 Stop bit 0x01: 2 Stop bits	yes

No.	Address	Parameter	Default Value	Description	SW Reset required
27	006F	UartSpeed	0x03	Speed of the Hardware UART interface. 2400: 0x00 4800: 0x01 7200: 0x02 9600: 0x03 19200: 0x04 38400: 0x05 57600: 0x06 115200: 0x07 230400: 0x08 460800: 0x09 921600: 0x0A	yes
28	0070-00AE	RemoteDe- vices	0x00..0x00	Default connections database, to be connected during boot-up or by sending a command.	no
29	00AF	-	0xFF	Reserved	-
30	00B0	-	0xFF	Reserved	-
31	00B1	-	0xFF	Reserved	-
32	00B2-00B3	-	0xFFFF	Reserved	-
33	00B4	-	0xFF	Reserved	-
34	00B5-00B8	-	0xFFFF	Reserved	-
35	00B9-00BC	-	0xFFFF	Reserved	-
36	00BD-00EE	Filler 1	0xFF ... 0xFF	Filler (not used)	
37	00EF-011 E	Service- Records	0xFF..0xFF	SDP/Security info storage	no
38	011 F-0346	CoreNvs- LinkKeys	0xFF..0xFF	Link key storage (24 keys)	no
39	0347-0355	-	0xFF..0xFF	Reserved	
40	0356-0367	-	0xFF..0xFF	Reserved	
41	0368-037F	Filler2	0xFF..0xFF	Filler, not used	
42	0380-137F	SdpRecord s	0xFF..0xFF	SDP record storage	no
43	1380-1 FFF	-	0xFF..0xFF	Reserved	-

Automatic operation on/off

The Automatic operation flag is checked after Reset/Boot-up and on incoming links. According to this flag the firmware will initiate automatic steps.

Automatic Operation On:

- ☐ Connect to "Default connections stored in NVS"
- ☐ Switch UART to "Transparent Mode" after first incoming link ("Transparent Slave")

Automatic Operation Off:

- ☐ No automatic steps after Reset/Boot-up

Force Master on/off

This flag improves multipoint operation of the RBT-001. In normal operation, the RBT-001 will be slave for any incoming link, which limits the number of incoming connects to 2 devices.

In case the Force Master Bit in the NVS is set to On, the RBT-001 will request a role switch on any incoming link to become Master.

Force Master On:

- ☐ Request Role Switch on incoming link to become Bluetooth Master. Link will be dropped if role switch not successful. Force Master Off:
- ☐ Accept incoming link as slave

Default connections stored in NVS

In case the automatic operation bit is set to On, the default connections database is checked for any valid entry. If a valid entry is found, the RBT-001 tries three times to connect to each device stored in the database.

Default connection transparent On/Off

Each default connection can be configured to switch the UART to "Transparent Mode" or to "Command Mode" after link establishment. In case the transparent flag is set, the RBT-001 will switch to "Transparent Mode" once the link is established.

NOTE: The transparent flag can only be activated if only one default connection is stored in the database

Operation States

As combination out of the different parameters and usage scenarios, the following states can be defined for the RBT-001. The parameters and possibilities for this states are summarized in the following Table "Overview of Operation States".

An example on how to read the Table: E.g. Scatternet Master (assuming default NVS settings).

- ☐ In Scatternet Master, the device is in command mode (listening to commands).
- ☐ The automatic flag in NVS has no influence. The device is discoverable and connectable for other devices.
- ☐ The device is Master for x slaves and slave to 1 Master.
- ☐ It is possible to search for other devices but it is not possible to be connected from another device. RAW data traffic is not possible.
- ☐ No Incoming link possible.
- ☐ In case the command interface is used to establish a link to another device, the device stays in "Scatternet Master".
- ☐ A UART BREAK has no influence on the functionality. Sending the command "Transparent Mode" is not applicable for this mode.

Parameter / State	Idle Automatic	Idle Non-Automatic	Piconet Master	Scatternet Master		Transparent Master	Single Slave	Scatternet Slave	Transparent Slave
				Command	Command				
UART Mode	Command	Command	Command	Command	Command	Transparent	Command	Command	Transparent
Automatic Operation	0x01 (On)	0x00 (Off)	-	-	-	-	-	-	-
Discoverable ¹	yes	yes	yes	yes	yes	no	yes	no	no
Connectable ²	yes	yes	yes	yes	no	no	yes	no	no
Bluetooth Role	-	-	Master to x Slaves ³	Master to x Slaves ⁴	Slave to 1 Master	Master to 1 Slave	Slave to 1 Master	Slave to 2 Masters	Slave to 1 Master
Possible to search for devices (Inquiry)	yes	yes	yes	yes	yes	no	yes	no	no
Connect to remote devices (actively)	yes	yes	yes ⁵	yes ⁴	yes ⁴	no	yes ⁴	no	no
Send Raw Data	-	-	no	no	no	yes	no	no	yes
State after incoming link	Transparent Slave	Single Slave	Scatternet Master	-	-	-	Scatternet Slave	-	-
State after outgoing link	Piconet Master	Piconet Master	Piconet Master	Scatternet Master	Scatternet Master	-	Scatternet Master	-	-
State after sending "Transparent Mode" ⁶	-	-	Transparent Master	-	-	-	Transparent Slave	-	-
State after UART BREAK	-	-	-	-	-	Piconet Master	-	-	Single Slave

1. Depending on parameter #14 in the non-volatile storage (default setting assumed)
2. Depending on parameter #15 in the non-volatile storage (default setting assumed)
3. With a value of $1 \leq x \leq 7$
4. With a value of $1 \leq x \leq 7$
5. Maximum number of slaves is 7
6. Transparent Mode can only be activated, if one and only one SPP link is active.

Idle, Automatic and Idle, Non-automatic

After boot-up, reset or after successfully releasing the last link, the RBT-001 stays within one of the Idle states.

In case the Automatic Operation bit is set to "On", it is also called "Idle, Automatic". Otherwise, the state is called "Idle, Non-Automatic"

In both modes, the device is able to accept incoming links or the application can use the command interface to actively inquire or establish a link.

The difference between Automatic "On" and "Off" just shows up at an incoming connection:

Incoming Link at Idle Automatic

In case, the RBT-001 is not connected to any other device and gets an incoming link request, it will

- ☐ ask for authentication or pin code exchange
- ☐ accept the link
- ☐ notify the application by an indicator
- ☐ turn off scanning (disabling page scan and inquiry scan)
- ☐ switch UART to "Transparent Mode" (1.3.2.2))

"Idle Automatic" is optimised for cable replacement applications, not requiring any interaction with the RBT-001 to accept an incoming link. As the RBT-001 automatically switches to transparent mode, the application can start sending Raw data immediately after receiving the notification.

Incoming Link in Idle Non-Automatic

In case, the RBT-001 is not connected to any other device and gets an incoming link request, it will

- ☐ ask for authentication or pin code exchange
- ☐ accept the link
- ☐ notify the application by an indicator
- ☐ change state to "Single Slave"

After link establishment, the command interface is still active, delivering incoming data using the "Incoming Data Indicator". To send data the command "SPP Send Data" needs to be used.

"Idle Non-automatic" is optimised for multi-profile applications which need to manage multiple links or different profiles at the same time. As the RBT-001 stays in command mode, the application still has full control over the RBT-001, to establish links or do configurations.

In case "Transparent Mode" would be beneficial for some data transfer, it can be reached anytime by sending the "Transparent Mode" command.

Piconet Master

In Piconet Master, the RBT-001 is in a link with one or more devices as Bluetooth Master. The Bluetooth specification describes the Master as the controlling device for the piconet. The Master defines the hopping sequence and manages the connection to each slave.

In general, a Bluetooth device will be master of the link, when it initiated the link (using the paging procedure). A device accepting an incoming link is called slave. However, every

Bluetooth device is able to request to a change of role (also called “role switch”) during link setup, therefore this general rule may not apply in 100% of the cases.

The RBT-001 can be assumed as master of the Bluetooth link and to be in “Piconet Master” when

- ☐ the command interface has been used to actively establish one or more links by using one of the following commands
 - “Establish Link”
 - “Connect to Default Connection”
 - a default connection has been established after power-up or “Reset”
- ☐ the RBT-001 accepted an incoming link while the Force Master bit in NVS was set to 0x01 (“Force Master Role”)

After successfully establishing a link, the RBT-001 will stay in “Command Mode”

The benefit of being master instead of slave is, that the RBT-001 is actively managing the link to all devices, so can assign each device the bandwidth it requires. With this, the RBT-001 is able to support up to 7 active links.

Scatternet Master

In case a Bluetooth device is master for one or several slaves and in parallel slave to one master, the connection scenario is called “scatternet”. The RBT-001 is able to be master to one or multiple slaves and in addition can be slave to maximum one master. Within the RBT-001 scenario, this state is called “Scatternet Master”.

The RBT-001 can be assumed to be in this state, after one of the following connection situations

- ☐ The RBT-001 accepts an incoming link as “Piconet Master” while the Force Master bit is set to 0x00.
- ☐ The device is “Single Slave” and actively establishes a link to another device by using one of the following commands
 - “SDAP Connect”
 - “Establish Link”
 - “Connect to Default Connection”

Once Scatternet Master has been reached, RBT-001 is not able to accept another incoming link. However the device will still be discoverable and will still answer to service requests.

Transparent Master

In case the RBT-001 is only connected to one other device (“point-to-point” connection), it might be beneficial for the application to send data directly to the UART interface, without having to use the command “SPP Send Data”. For this the RBT-001 offers the so call “Transparent Mode” on the UART, which allows to send data directly.

Since the RBT-001 in this case gets no information to which port to send this data to, transparent mode is only allowed on a point-to-point connection. Data will be routed directly from the UART interface to the remote Bluetooth device.

Transparent Master means, the RBT-001 is master for the point-to-point connection to one other link and Transparent Mode is switched on. This state is reached by one of the following situations

- ☐ Sending the “Transparent Mode” command in “Piconet Master” state.
- ☐ In case Automatic operation is ON, Default Connection setup after Reset, in which the transparent flag is set to 0x01.

- ❑ Default Connection setup by using "Connect to Default Connection", in which the transparent flag is set to 0x01. Leaving Transparent Master, initiated by a UART break, will lead into "Piconet Master" state.

Since the RBT-001 can not send any events or react on incoming commands, scanning is switched off and therefore the RBT-001 is not discoverable or connectable for other devices.

Single Slave

The Bluetooth specification defines a Bluetooth slave as the device which is connected by another device and adjusting to the timing of that device (Master). The slave synchronizes to the clock of Master and to its hopping sequence. In an active link, the master polls each slave (by default every 40 slots) to keep them synchronized but also to allow the slave to send data.

By default, the RBT-001 accepts any incoming link. Depending on the configuration of Security level ("Set Security Mode") and the Service Database entry, the device will ask for authentication.

The RBT-001 can be assumed to be in Single Slave after one of the following actions appeared:

- ❑ The RBT-001 accepted an incoming link and reports it by the "SPP Link Established Indicator", while the Automatic Operation flag is set to 0x00 (Non-automatic).
- ❑ The host sends a UART Break to a RBT-001 in "Transparent Slave"

In theory, there's also the possibility for Single Slave, in case the RBT-001 actively establishes a link to another device, which requests a role switch. In this case the initiating device will be slave. Since the role switch is not reported to the command interface and therefore can't be proven, this scenario shall not be discussed in this document.

In Single slave the RBT-001 still listens to commands on the UART. The device will be discoverable and connectable for other devices.

Scatternet Slave

An advanced but not most efficient connection state is the Scatternet Slave. In this mode the RBT-001 is slave to two different masters. This means, the RBT-001 needs to switch between two different synchronization states over time, serving each Master only for a limited time. While it is synchronized to Master 1 it is not able to listen to Master 2, therefore might miss the poll packages.

The RBT-001 can be assumed to be in Scatternet Slave after the following action appeared:

- ❑ The RBT-001 accepted an incoming link as "Single Slave", reported by "SPP Link Established Indicator".

The RBT-001 is able to manage such a link without issues. However, since the switching between two piconets consumes significant bandwidth, "Piconet Master" or at least "Scatternet Master" should be used instead, which for example can be reached by setting the Force Master Flag within the NVS ("Force Master Role").

In Scatternet Slave, discoverability and connectability are switched off.

Transparent Slave

A typical scenario for the RBT-001 is the cable replacement, in which the RBT-001 just waits for an incoming connection and the host connected over UART starts transmitting

data after an incoming link has been established. Since in many cases the RBT-001 just replaces a former cable connection, the "Transparent Mode" on the UART allows implementation without any software change on the data transmissions.

Transparent Slave means the device is slave on the Bluetooth link and the UART is switched to "Transparent Mode".

The RBT-001 can be assumed to be in "Transparent Slave" state after one of the following actions appeared:

- ☐ the RBT-001 accepted an incoming link in "Idle, Automatic", in which the Automatic Flag is switched to On.
- ☐ the host sent command "Transparent Mode" while the RBT-001 is in "Single Slave" state. Leaving Transparent Slave, initiated by a UART break, will lead into "Single Slave" state.

Since the RBT-001 can not send any events or react on incoming commands, scanning is switched off and therefore the RBT-001 is not discoverable or connectable for other devices.

UART Protocol Principles

The RBT-001 can be controlled with simple commands on the UART interface. The commands have to be sent within a special package format. The following sections describe the format of the command set packages.

Framing

The connection is considered “Error free”. But for packet recognition and synchronization, some framing is used. All packets sent in both directions are constructed after the following model:

Start delimiter	Packet Type identification	Op code	Data length	Check-sum	Packet Data	End delimiter
1 byte	1 byte	1 byte	2 bytes	1 byte	<Data length> bytes	1 byte
----- Checksum -----						

Start delimiter

The start delimiter indicates the RBT-001 the beginning of a new package. The “STX” char is used as start delimiter: STX = 0x02

Packet type identification

This byte identifies the type of packet. The following types are valid:

Code	Packet Type	Description
0x52 'R'	Request (REQ)	A request sent to the Bluetooth module. All request are answered by exactly one confirm.
0x43 'C'	Confirm (CFM)	The Bluetooth modules confirm to a request. All request are answered by exactly one confirm.
0x69 'i'	Indication (IND)	Information sent from the Bluetooth module, that is not a direct confirm to a request.
0x72 'r'	Response (RES)	An optional response to an indication. This is used to respond to some type of indication messaged.

All other values are reserved.

Opcode

The opcode is a command specifier. Each command is represented by this one byte identifier, as in the following table.

Opcode	Value
GAP_INQUIRY	0x00
GAP_DEVICE_FOUND	0x01

Opcode	Value
GAP_REMOTE_DEVICE_NAME	0x02
GAP_READ_LOCAL_NAME	0x03
GAP_WRITE_LOCAL_NAME	0x04
GAP_READ_LOCAL_BDA	0x05
GAP_SET_SCANMODE	0x06
GAP_GET_FIXED_PIN	0x16
GAP_SET_FIXED_PIN	0x17
GAP_GET_PIN	0x75
GAP_GET_SECURITY_MODE	0x18
GAP_SET_SECURITY_MODE	0x19
GAP_REMOVE_PAIRING	0x1B
GAP_LIST_PAIRIED_DEVICES	0x1C
GAP_ENTER_SNIFF_MODE	0x21
GAP_EXIT_SNIFF_MODE	0x37
GAP_ENTER_PARK_MODE	0x38
GAP_EXIT_PARK_MODE	0x39
GAP_ENTER_HOLD_MODE	0x3A
GAP_SET_LINK_POLICY	0x3B
GAP_GET_LINK_POLICY	0x3C
GAP_POWER_SAVE_MODE_CHANGED	0x3D
GAP_ACL_ESTABLISHED	0x50
GAP_ACL_TERMINATED	0x51
SPP_SET_PORT_CONFIG	0x07
SPP_GET_PORT_CONFIG	0x08
SPP_PORT_CONFIG_CHANGED	0x09
SPP_ESTABLISH_LINK	0x0A
SPP_LINK_ESTABLISHED	0x0B
SPP_INCOMMING_LINK_ESTABLISHED	0x0C
SPP_RELEASE_LINK	0x0D
SPP_LINK_RELEASED	0x0E
SPP_SEND_DATA	0x0F
SPP_INCOMING_DATA	0x10
SPP_TRANSPARENT_MODE	0x11
SPP_CONNECT_DEFAULT_CON	0x12
SPP_STORE_DEFAULT_CON	0x13
SPP_GET_LIST_DEFAULT_CON	0x14
SPP_DELETE_DEFAULT_CON	0x15
SPP_SET_LINK_TIMEOUT	0x57
SPP_GET_LINK_TIMEOUT	0x58
SPP_PORT_STATUS_CHANGED	0x3E
SPP_GET_PORT_STATUS	0x40

Opcode	Value
SPP_PORT_SET_DTR	0x41
SPP_PORT_SET_RTS	0x42
SPP_PORT_BREAK	0x43
SPP_PORT_OVERRUN_ERROR	0x44
SPP_PORT_PARITY_ERROR	0x45
SPP_PORT_FRAMING_ERROR	0x46
SDAP_CONNECT	0x32
SDAP_DISCONNECT	0x33
SDAP_CONNECTION_LOST	0x34
SDAP_SERVICE_BROWSE	0x35
SDAP_SERVICE_SEARCH	0x36
SDAP_SERVICE_REQUEST	0x1E
SDAP_ATTRIBUTE_REQUEST	0x3F
CHANGE_NVS_UART_SPEED	0x23
CHANGE_UART_SETTINGS	0x48
SET_PORTS_TO_OPEN	0x22
GET_PORTS_TO_OPEN	0x1F
RESTORE_FACTORY_SETTINGS	0x1A
STORE_CLASS_OF_DEVICE	0x28
FORCE_MASTER_ROLE	0x1D
READ_OPERATION_MODE	0x49
WRITE_OPERATION_MODE	0x4A
SET_DEFAULT_LINK_POLICY	0x4C
GET_DEFAULT_LINK_POLICY	0x4D
SET_EVENT_FILTER	0x4E
GET_EVENT_FILTER	0x4F
SET_DEFAULT_LINK_TIMEOUT	0x55
GET_DEFAULT_LINK_TIMEOUT	0x56
SET_DEFAULT_LINK_LATENCY	0x63
GET_DEFAULT_LINK_LATENCY	0x64
SET_PCM_SLAVE_CONFIG	0x74
ENABLE_SDP_RECORD	0x29
DELETE_SDP_RECORDS	0x2A
STORE_SDP_RECORD	0x31
RESET	0x26
RBT-001_READY	0x25
TEST_MODE	0x24
WRITE_ROM_PATCH	0x47
READ_RSSI	0x20
RF_TEST_MODE	0x4B
DISABLE_TL	0x52

Opcode	Value
TL_ENABLED	0x53
AWAIT_INITIALIZATION_EVENT	0x66
ENTER_BLUETOOTH_MODE	0x66
READ_NVS	0x72
WRITE_NVS	0x73

Data length

Number of bytes in the “Packet data” area. The maximum size is 333 bytes.

Packet data

The data fields hold binary data; hence both 0x02 (=STX) and 0x03 (=ETX) are allowed as data.

Checksum

This is a simple Block Check Character (BCC) checksum of the bytes from “Packet type” to, and including, “data length”. The BCC checksum is calculated as the low byte of the sum of all bytes.

E.g. if the sum of all bytes are 0x3724, the checksum is 0x24.

End delimiter

The “ETX” char is used as end delimiter: ETX = 0x03

Retransmission

The connection is considered “Error free”, hence no need for implementing time-outs and retransmissions.

Flow control

A transparent data-mode is supported for RFCOMM communication. When using this transparent mode, full hardware handshake is advised.

When not in transparent mode, the protocol principle of REQ-CFM, limits the need of buffer capacity. As IND's can come out of REQ-CFM sequence, and is unconfirmed, the user device has to be able to read these data fast enough / have enough buffer capacity.

Byte Order

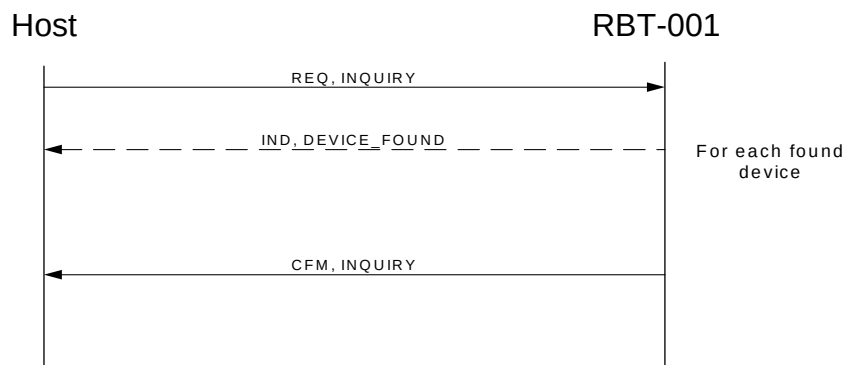
The byte order of the protocol is Little Endian, if nothing else is specified.

Commands in Command Mode

The RBT-001 implements a complete command set for bluetooth operation and local configuration. The command set is based on a request/confirm scheme meaning any command will be confirmed by an appropriate event including the same opcode.

Searching for remote devices

The RBT-001 implements the basic Bluetooth functionalities ("GAP"), and offers simple interfaces for locating other Bluetooth devices (Inquiry and name discovery), and easy configuration of the local Bluetooth device.



Inquiry

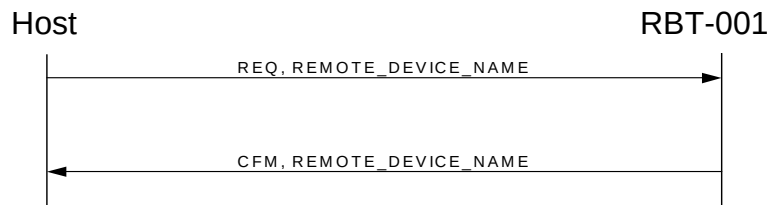
Description	Initiates a search for other Bluetooth devices.	
PacketType	REQ	
Opcode	GAP_INQUIRY	
DataLength	3	
Data	Length 1 byte	Duration of inquiry Range: 0x01 -0x30 (1.28s - 61.44s)
	NumResponses 1 byte	Maximum number of responses Range: 0x00 - 0xFF 0x00 = Unlimited number of responses.
	Mode 1 Byte	General Inquiry 0x00 Limited Inquiry 0x01

Description	Confirms that the search for other Bluetooth devices is complete.	
PacketType	CFM	
Opcode	GAP_INQUIRY	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_DURATION_OUT_OF_RANGE ERROR_INVALID_MODE ERROR_INVALID_NO_OF_PARAMETERS

Device Found

Description	Indicates that a device has been found.	
PacketType	IND	
Opcode	GAP_DEVICE_FOUND	
DataLength	9	
Data	BdAddr 6 bytes	Bluetooth device address of the found device.
	DeviceClass 3 byte	Class of the found device.

Getting user-friendly name of remote Bluetooth Device

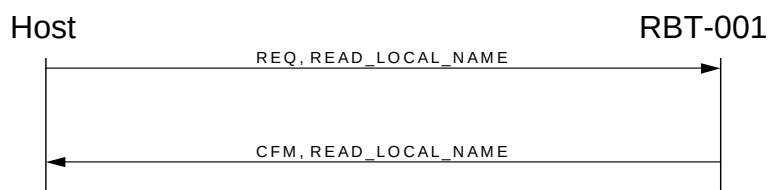


Remote Device Name

Description	Request the user-friendly name from a known remote Bluetooth device.	
PacketType	REQ	
Opcode	GAP_REMOTE_DEVICE_NAME	
DataLength	6	
Data	BdAddr 6 byte	Bluetooth device address for the remote device

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_REMOTE_DEVICE_NAME	
DataLength	8+ NameLength if ok, otherwise 8	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_TIMEOUT
	BdAddr 6 byte	Bluetooth device address for the remote device
	NameLength 1 byte	Number of bytes in device name
	DeviceName Length bytes	The user-friendly name of the remote device. NULL terminated. Maximum length is 40 bytes.

Reading the user-friendly name of the local device

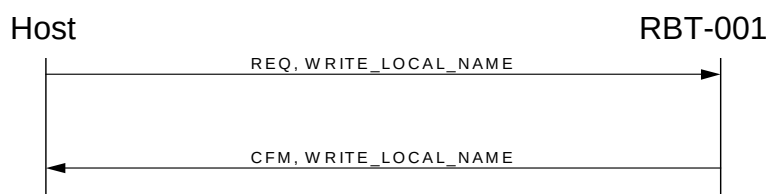


Read Local Name

Description	Request the user-friendly name for the local Bluetooth device.	
PacketType	REQ	
Opcode	GAP_READ_LOCAL_NAME	
DataLength	0	
Data	None	

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_READ_LOCAL_NAME	
DataLength	2 + NameLength	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_UNKNOWN_ERROR
	NameLength 1 byte	Number of bytes in device name
	DeviceName NameLength bytes	The user-friendly name of the local device. The string is NULL terminated. . Max length is 40 bytes.

Changing the user-friendly name of the local device

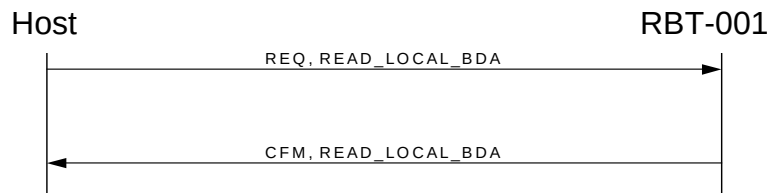


Write Local Name

Description	Change the user-friendly name for the local Bluetooth device. The name is stored in NVS	
PacketType	REQ	
Opcode	GAP_WRITE_LOCAL_NAME	
DataLength	1+ NameLength	
Data	NameLength	Number of bytes in device name
	DeviceName Length bytes	The user-friendly name of the local device. (String must be NULL terminated). Max length is 40 bytes.

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_WRITE_LOCAL_NAME	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_NAME_TOO_LONG ERROR_INVALID_NO_OF_PARAMETERS

Reading the Bluetooth device address of the local device



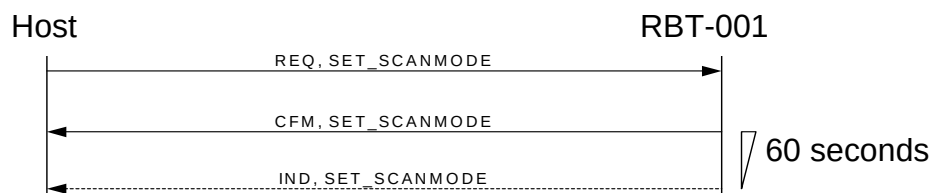
Read Local Bluetooth Address

Description	Read the Bluetooth device address of the local Bluetooth device.	
PacketType	REQ	
Opcode	GAP_READ_LOCAL_BDA	
DataLength	0	
Data	None	

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_READ_LOCAL_BDA	
DataLength	7	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_UNKNOWN_ERROR
	BdAddr 6 byte	Bluetooth device address for the local device

Changing Scanmode (Connectability and Discoverability)

Connectability and Discoverability is set with this single request.



Set Scan Mode

Description	Change the Bluetooth scan mode. Automatic limited discoverable mode automatically toggles between general and limited inquiry scanning. This mode defined by the Bluetooth GAP profile specification, refer to part K.1, section 6.2.1. The automatic Limited discoverable mode times out after 60 sec. At this point the RBT-001 sends the GAP_SET_SCANMODE indication and resets page and inquiry scan settings to the value that was stored before the automatic limited discoverable mode was entered. The Connectability mode and discoverability modes are stored in NVS and restored during startup. An exception are the limited discoverable mode and automatic limited discoverable modes are selected, in this case neither connectability mode or discoverability mode are stored in NVS.	
PacketType	REQ	
Opcode	GAP_SET_SCANMODE	
DataLength	2	
Data	Connectability 1 byte	0x00 = Not connectable 0x01 = Connectable 0x81 = Connectable using Interlaced Scanning.
	Discoverability 1 byte	0x00 = Non discoverable 0x01 = General discoverable 0x81 = General discoverable using Interlaced Scanning. 0x02 = Limited discoverable 0x82 = Limited discoverable using Interlaced Scanning. 0x03 = Automatic limited discoverable mode 0x83 = Automatic limited discoverable mode using Interlaced Scanning.

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_SET_SCANMODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_CONNECTABILITY_PARAMETER ERROR_INVALID_DISCOVERABILITY_PARAMETER ERROR_INVALID_NO_OF_PARAMETERS ERROR_UNKNOWN_ERROR

Set Scan Mode Indication

Description	Indication send from the device, when automatic limited discoverable mode has ended.	
PacketType	IND	
Opcode	GAP_SET_SCANMODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNKNOWN_ERROR

Bluetooth Security

Bluetooth security is part of the Generic Access Profile GAP. It is controlled by:

- ❑ Security Mode
 - o Security Mode 1:
 - No Security, the device never will ask for authentication or pairing.
 - o Security Mode 2:
 - The level of security (Authorization, Authentication, Encryption) is determined by the setting in the service database entries. Each entry can have different security requirements.
 - no authentication necessary for SDAP links
 - o Security Mode 3:
 - Authentication already necessary on Link Manager level: SDAP links already require authentication (Service requests)
 - Device always asks for authentication
- ❑ Service Database Entry (only for Security Mode 2)
 - o Each entry can specify the settings for authentication and encryption

The RBT-001 by default is in Security Mode 2.

Get Fixed PIN

Description	Reads the current fixed pin code from NVS
PacketType	REQ
Opcode	GAP_GET_FIXED_PIN
DataLength	0

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_GET_FIXED_PIN	
DataLength	2+Pinlength	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	Pinlength 1 byte	Length of pin code, in bytes. 0x00: No Fixed pin, RBT-001 will send "Pin request indicator" in case a pin is required. Range: 0x01-0x10
	Pincode Pinlength bytes	PIN code used when the two Bluetooth devices are paired. The maximum length of a PIN code is 128 bits (16 bytes).

Set Fixed PIN

Description	Stores a new fixed pin code in NVS	
PacketType	REQ	
Opcode	GAP_SET_FIXED_PIN	
DataLength	1+ Pinlength	
Data	Pinlength 1 byte	Length of pin code, in bytes. Range: 0x01-0x10
	Pincode Pinlength bytes	PIN code used when the two Bluetooth devices are paired. The maximum length of a PIN code is 128 bits (16 bytes).

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_SET_FIXED_PIN	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_PINCODE_LENGTH

PIN request

Description	This event is used to inform the Host when a PIN code is requested during authentication of an ACL link. This event is only generated if the length of the Fixed pin parameter stored in the NVS is set to 0.	
PacketType	IND	
Opcode	GAP_GET_PIN	
DataLength	6	
Data	BdAddr 6 byte	The Bluetooth device address of the remote device.

Description	This command is used to send a PIN code to the RBT-001 module as response to a GAP_GET_PIN indication.	
PacketType	REQ	
Opcode	GAP_GET_PIN	
DataLength	1	
DataLength	7 + Pinlength	
Data	BdAddr 6 byte	The Bluetooth device address of the remote device.
	Pinlength 1 byte	Length of pin code, in bytes. Range: 0x00-0x10 0x00 indicates that the Host does not allow the authentication of the ACL link.
	Pincode Pinlength bytes	PIN code used when the two Bluetooth devices are paired. The maximum length of a PIN code is 128 bits (16 bytes).

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_GET_PIN	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_PINCODE_LENGTH

Get Security Mode

Description	Reads the current security mode of the Bluetooth device.	
PacketType	REQ	
Opcode	GAP_GET_SECURITY_MODE	
DataLength	0	
Data	None	

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_GET_SECURITY_MODE	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	Mode 1 byte	The current Bluetooth security mode. 0x01 Security mode 1 0x02 Security mode 2 0x03 Security mode 3 0x83 Security mode 3 with link level encryption

Set Security Mode

Description	Changes the current security mode of the Bluetooth device. The security mode is stored in NVS and restored during power up.	
PacketType	REQ	
Opcode	GAP_SET_SECURITY_MODE	
DataLength	1	
Data	Mode 1 byte	The current Bluetooth security mode. 0x01 Security mode 1 0x02 Security mode 2 0x03 Security mode 3 0x83 Security mode 3 with link level encryption

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	GAP_SET_SECURITY_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_INVALID_SECURITY_MODE

Pairing

The pairing process is part of the authentication procedure. If a local or remote service asks for authentication during link establishment, the authentication process will check if a link key already exists between the two devices. If not, the Link Manager initiates the pairing process. Within this, the two devices exchange a PIN code and create a secure link key which will be stored in each device.

During next link setup, the authentication routine takes the existing link key and proceeds without this pairing procedure.

The RBT-001 has a fixed PinCode which can be changed with the command “Change fixed Pin”. This pin is used during any pairing procedure. In case the stored PinCode has length 0x00, the RBT-001 will request a Pin from the application using the “Pin Request” indicator. The application needs to respond with the appropriate pin in the Pin request.

Remove Pairing

Description	Remove pairing with a remote device.	
PacketType	REQ	
Opcode	GAP_REMOVE_PAIRING	
DataLength	6	
	BdAddress 6 byte	Remove pairing to the BdAddress.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_REMOVE_PAIRING	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_LINKKEY_DOES_NOT_EXISTS

List Paired Devices

Description	Request a list of paired devices from NVS	
PacketType	REQ	
Opcode	GAP_LIST_PAIRIED_DEVICES	
DataLength	0	

Description	Response to the request above.	
PacketType	CFM	
Opcode	LIST_PAIRIED_DEVICES	
DataLength	2 +6 * DeviceCount	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	DeviceCount 1 byte	Number of devices in the list of paired devices If 0 the device is not paired to any other devices. The maximum number of paired devices is 7.
	BdAddresses 6 byte * DeviceCount	The list of paired devices

"Air interface" Low Power Modes

A Bluetooth link is based on a physically synchronized connection, which means that the devices can only communicate after successful synchronization. For this, each package also includes some synchronization information. Also a specific polling scheme is in place to keep synchronization if no traffic is necessary.

As the slave has to actively listen to packages from the master, there are different methods to decrease the necessary active receive slots on devices.

The ability to switch to those specific modes is controlled by the Link Policy. To make sure both devices support the low power mode requested, Link Policy can be set first. It will only be successful if both sides support it.

Enter Sniff Mode

Command to enter the sniff mode. The command includes the maximum and minimum value for the sniff interval. After sending the command, Master and slave will calculate a reasonable sniff time and will switch into Sniff mode.

Description	This command is used to request sniff mode on a given link with user specified parameters.	
PacketType	REQ	
Opcode	GAP_ENTER_SNIFF_MODE	
DataLength	14	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to put the link in sniff mode.
	SniffMaxInterval 2 bytes	Maximum sniff interval in slots.
	SniffMinInterval 2 bytes	Minimum sniff interval in slots.
	SniffAttempt 2 bytes	Number of slots the slave must listen, beginning at the sniff slot, even if it does not receive a packet with its own AM.
	SniffTimeout 2 bytes	Number of additional slots the slave must listen if it continues to receive only packets with its own AM address.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_ENTER_SNIFF_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION

Exit Sniff Mode

Description	This command is used to exit a current sniff mode on a given link.	
PacketType	REQ	
Opcode	GAP_EXIT_SNIFF_MODE	
DataLength	6	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to exit the current sniff mode.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_ENTER_SNIFF_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION

Enter Park Mode

In Park Mode the slave will lose its active member address and will not longer be part of the piconet. It will be kept synchronized by beacons within the specified interval range.

Description	This command is used to request park mode on a given link with user specified parameters.	
PacketType	REQ	
Opcode	GAP_ENTER_PARK_MODE	
DataLength	10	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to put the link in park mode.
	BeaconMaxInterval 2 bytes	Acceptable longest length of the interval between beacons.
	BeaconMinInterval 2 bytes	Acceptable shortest length of the interval between beacons.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_ENTER_PARK_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION

Exit Park Mode

Description	This command is used to exit a current park mode on a given link.	
PacketType	REQ	
Opcode	GAP_EXIT_PARK_MODE	
DataLength	6	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to exit the current park mode.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_ENTER_PARK_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION

Enter Hold Mode

Description	This command is used to request Hold mode on a given link with user specified parameters.	
PacketType	REQ	
Opcode	GAP_ENTER_HOLD_MODE	
DataLength	10	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to put the link in Hold mode.
	HoldMaxInterval 2 bytes	Maximum length of the Hold interval for which the Host may actually enter into the hold mode after negotiation with the remote device.
	HoldMinInterval 2 bytes	Minimum length of the Hold interval for which the Host may actually enter into the hold mode after the negotiation with the remote device.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_ENTER_HOLD_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION

Set Link Policy

Description	This command is used to change the current link policy setting for the given link.	
PacketType	REQ	
Opcode	GAP_SET_LINK_POLICY	
DataLength	8	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to change the link policy settings for the link.
	LinkPolicy 2 byte	Bitfield: 0x0001 = Master-slave switch allowed 0x0002 = Hold mode allowed 0x0004 = Sniff mode allowed 0x0008 = Park mode allowed

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_SET_LINK_POLICY	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION ERROR_ILLEGAL_LINK_POLICY

Get Link Policy

Description	This command is used to get the current link policy setting for the given link.	
PacketType	REQ	
Opcode	GAP_GET_LINK_POLICY	
DataLength	6	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of which to get the current link policy settings for the link.

Description	Response to the request above.	
PacketType	CFM	
Opcode	GAP_GET_LINK_POLICY	
DataLength	3	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION
	LinkPolicy 2 byte	Bitfield: 0x0001 = Master-slave switch allowed 0x0002 = Hold mode allowed 0x0004 = Sniff mode allowed 0x0008 = Park mode allowed

Power Save Mode Changed

Description	This indication is sent to the host when changes the power save mode on a link occur.	
PacketType	IND	
Opcode	GAP_POWER_SAVE_MODE_CHANGED	
DataLength	8	
Data	Status 1 byte	ERROR_OK ERROR_ATTEMPT_FAILED ERROR_UNSPECIFIED_ERROR
	BdAddr 6 bytes	The Bluetooth address of the remote device for which the power save mode has changed on the link.
	Mode 1 byte	0x00 = Active mode (Left power save mode) 0x01 = Hold mode (Hold mode entered) 0x02 = Sniff mode (Sniff mode entered) 0x03 = Park mode (Park mode entered)

Device Low Power Modes

In certain applications the RBT-001 will be used most of the time in a waiting status, meaning it is waiting for being connected or listening to commands. To reduce power consumption of the system, the RBT-001 supports a specific Wake up functionality.

The RBT-001 supports to disable the UART transport layer (command "Disable Transport Layer") to switch off the command interpreter and all hardware components not needed for the current operation. The interface can be reactivated again by either side by using hardware pins. The RBT-001 uses the RTS signal pin to wake up the host. The RTS / CTS signals are connected in a NULL-Modem fashion (i.e. "crossed"), meaning that RTS on the Host is connected to CTS on the RBT-001 and vice versa. Therefore the host would need to be able to monitor its CTS input or has to use a separate hardware pin. In case the RBT-001 has to be triggered by the host, the RTS pin is used as the hardware Wake-Up signal.

Disable Transport Layer

Description	This Command disables the transport layer and thereby allowing power saving.	
PacketType	REQ	
Opcode	DISABLE_TL	
DataLength	0	

Description	Response to the request above.	
PacketType	CFM	
Opcode	DISABLE_TL	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_COMMAND_DISALLOWED

Description	This indication is sent when the transport layer is re-enabled. Transport layer is re-enabled by pulling RTS signal of the host or an event is ready to be sent from the RBT-001 to the host, normally due to an air-interface event.	
PacketType	IND	
Opcode	TL_ENABLED	
DataLength	1	
Data	Status 1 byte	ERROR_OK

ACL indications

If the Event Filter is set to "Report all events", the RBT-001 indicates any established and terminated ACL link to the host. This feature enables the user to monitor the Bluetooth physical interface. In case the establishment failed the indicators report the reason for terminating or not establishing the link. The ACL error events can be found in Table "ACL Error Codes" on page 73.

ACL Established

Description	This indication is sent to the host when an ACL link is established.	
PacketType	IND	
Opcode	GAP_ACL_ESTABLISHED	
DataLength	7	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device.
	Status 1 byte	See "ACL Error Codes" on page 73.

ACL Terminated

Description	This indication is sent to the host when an ACL link is terminated.	
PacketType	IND	
Opcode	GAP_ACL_TERMINATED	
DataLength	8	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device.
	Status 1 byte	See "ACL Error Codes" on page 73.

Serial port profile (SPP)

An active SPP link appears as a virtual serial port connection between two devices. As any other serial connection it has different settings for that “virtual” serial port. The following settings enable the host to change specific port settings on that virtual port. The reference for all commands is the local RFCOMM port the link has been set up with.

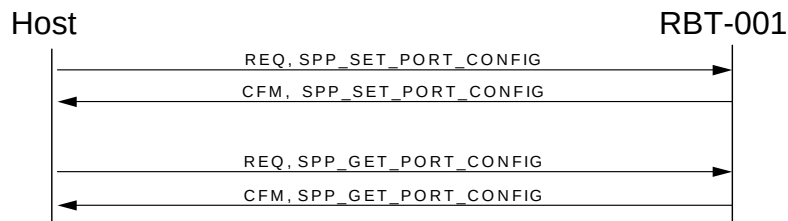
All commands in this section are only handled at application level and have no direct impact on the Bluetooth link.

The RBT-001 implements the serial port profile (SPP), and offers a very simple interface for RFCOMM communication.

Open / Close an RFCOMM port

The device will automatically open all configured "ports to open" at start up.

Reading / changing current port configuration



Set Port Config

Description	Write the configuration for the SPP port. These are “virtual” settings for the air connection, not the settings for the RBT-001 UART. Note: The baudrate in this configuration has no impact on the throughput on the Bluetooth link itself or the RBT-001. It is not used by the RFCOMM layer.			
PacketType	REQ			
Opcode	SPP_SET_PORT_CONFIG			
DataLength	6			
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30		
	BaudRate 1 byte	Baudrate 0x00 = 2400 baud 0x01 = 4800 baud 0x02 = 7200 baud 0x03 = 9600 baud 0x04 = 19200 baud 0x05 = 38400 baud 0x06 = 57600 baud 0x07 = 115200 baud 0x08 = 230400 baud Note: The baudrate in this configuration has no impact on the throughput on the Bluetooth link itself or the RBT-001. It is not used by the RFCOMM layer. The parameter might be used as informative parameter on the application level.		
	Portsettings 1 byte	Bitfield: 000000XX	Number of databits	00=5 bits 01=6 bits 02=7 bits 03=8 bits
		Bitfield: 00000X00	Number of stopbits	0 = 1 1 = 1.5
		Bitfield: 0000X000	Parity	0 = No parity 1 = Parity

		Bitfield: 00XX0000	ParityType	00=ODD 01=EVEN 02=MARK 03=SPACE
		Bitfield: XX000000	Reserved	
	FlowControl 1 byte	Bitfield: 0000000X	XonXoffOnInput	0=Disable 1=Enable
		Bitfield: 000000X0	XonXoffOnOutput	0=Disable 1=Enable
		Bitfield: 00000X00	RtrOnInput	0=Disable 1=Enable
		Bitfield: 0000X000	RtrOnOutput	0=Disable 1=Enable
		Bitfield: 000X0000	RtcOnInput	0=Disable 1=Enable
		Bitfield: 00X00000	RtcOnOutput	0=Disable 1=Enable
		Bitfield: XX000000	Reserved	
	XonChar 1 byte	Char used for Xon, if Xon/Xoff flowcontrol is used.		
	XoffChar 1 byte	Char used for Xoff, if Xon/Xoff flowcontrol is used.		

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_SET_PORT_CONFIG	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_UNSPECIFIED_ERROR ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN ERROR_UART_SPEED_OUT_OF_RANGE
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Get Port Config

Description	Read the configuration for the SPP port. This is "virtual" settings for the air connection, not settings for the RBT-001 serial port.	
PacketType	REQ	
Opcode	SPP_GET_PORT_CONFIG	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_GET_PORT_CONFIG	
DataLength	7	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN

	LocalPort 1 byte	Local RFCOMM port number. Range 1-30		
	BaudRate 1 byte	Baudrate 0x00 = 2400 baud 0x01 = 4800 baud 0x02 = 7200 baud 0x03 = 9600 baud 0x04 = 19200 baud 0x05 = 38400 baud 0x06 = 57600 baud 0x07 = 115200 baud 0x08 = 230400 baud		
	Portsettings 1 byte	Bitfield: 000000XX	Number of databits	00=5 bits 01=6 bits 02=7 bits 03=8 bits
		Bitfield: 00000X00	Number of stopbits	0 = 1 1 = 1.5
		Bitfield: 0000X000	Parity	0 = No parity 1 = Parity
		Bitfield: 00XX0000	ParityType	00=ODD 01=EVEN 02=MARK 03=SPACE
		Bitfield: XX000000	Reserved	
	FlowControl 1 byte	Bitfield: 0000000X	XonXoffOnInput	0=Disable 1=Enable
		Bitfield: 000000X0	XonXoffOnOutput	0=Disable 1=Enable
		Bitfield: 00000X00	RtrOnInput	0=Disable 1=Enable
		Bitfield: 0000X000	RtrOnOutput	0=Disable 1=Enable
		Bitfield: 000X0000	RtcOnInput	0=Disable 1=Enable
		Bitfield: 00X00000	RtcOnOutput	0=Disable 1=Enable
		Bitfield: XX000000	Reserved	
	XonChar 1 byte	Char used for Xon, if Xon/Xoff flowcontrol is used.		
	XoffChar 1 byte	Char used for Xoff, if Xon/Xoff flowcontrol is used.		

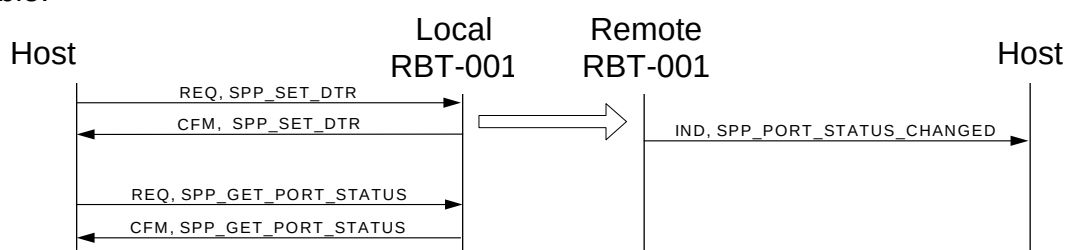
Port Config Changed

Description	Sent from the RBT-001 when remote device has changed the port configuration.			
PacketType	IND			
Opcode	SPP_PORT_CONFIG_CHANGED			
DataLength	6			
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30		
	BaudRate 1 byte	Baudrate 0x00 = 2400 baud 0x01 = 4800 baud 0x02 = 7200 baud 0x03 = 9600 baud 0x04 = 19200 baud 0x05 = 38400 baud 0x06 = 57600 baud 0x07 = 115200 baud 0x08 = 230400 baud		
	Portsettings 1 byte	Bitfield: 000000XX	Number of databits	00=5 bits 01=6 bits 02=7 bits 03=8 bits
		Bitfield: 00000X00	Number of stopbits	0 = 1 1 = 1.5

		Bitfield: 0000X000	Parity	0 = No parity 1 = Parity
		Bitfield: 00XX0000	ParityType	00=ODD 01=EVEN 02=MARK 03=SPACE
		Bitfield: XX000000	Reserved	
FlowControl	1 byte	Bitfield: 0000000X	XonXoffOnInput	0=Disable 1=Enable
		Bitfield: 000000X0	XonXoffOnOutput	0=Disable 1=Enable
		Bitfield: 00000X00	RtrOnInput	0=Disable 1=Enable
		Bitfield: 0000X000	RtrOnOutput	0=Disable 1=Enable
		Bitfield: 000X0000	RtcOnInput	0=Disable 1=Enable
		Bitfield: 00X00000	RtcOnOutput	0=Disable 1=Enable
		Bitfield: XX000000	Reserved	
XonChar	1 byte	Char used for Xon, if Xon/Xoff flowcontrol is used.		
XoffChar	1 byte	Char used for Xoff, if Xon/Xoff flowcontrol is used.		

Reading / changing current port status

SPP link allows signaling of Modem status and line status over the Bluetooth link. The commands for changing Modem / Line status will only have effect if the SPP link is available.



Get Port Status

Description	Get the current state of the modem status and line status. This command resets the value (to 0) of the following members of the port status: OverrunError ParityError FramingError BreakLength The value of DSR and CTS are only changed when new values are received from the remote device!	
PacketType	REQ	
Opcode	SPP_GET_PORT_STATUS	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Confirm to the request above.			
PacketType	CFM			
Opcode	SPP_GET_PORT_STATUS			
DataLength	5			
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN		
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30		
	PortStatus 1 byte	Bitfield: 0000000X	DTR	0 = Low 1 = High
		Bitfield: 000000X0	RTS	0 = Low 1 = High
		Bitfield: 00000X00	DSR	0 = Low 1 = High
		Bitfield: 0000X000	CTS	0 = Low 1 = High
		Bitfield: 000X0000	Overrun Error	0 = No Error 1 = Overrun Error
		Bitfield: 00X00000	Parity Error	0 = No Error 1 = Parity Error
		Bitfield: 0X000000	Framing Error	0 = No Error 1 = Framing Error
		Bitfield:X0000000	DLC established	0 = No DLC 1 = DLC is available
	Break Length 2 bytes	The length in ms of the detected break. The value 0 is used to indicate that no break has been detected.		

Set DTR

Description	This command sets the state of the DTR bit. Since RFCOMM acts as a "null modem" where DTR and DSR are connected, the remote device will see this as a change of the state of the DSR signal. In case the remote device is a RBT-001, this remote RBT-001 will send a "SPP Port Status Changed Indicator" to its application.		
PacketType	REQ		
Opcode	SPP_PORT_SET_DTR		
DataLength	2		
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30	
	State	0: False. 1: True.	

Description	Confirm to the request above.		
PacketType	CFM		
Opcode	SPP_PORT_SET_DTR		
DataLength	2		
Data	Status 1 byte	ERROR_OK ERROR_ILLEGAL_STATE_VALUE ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN	
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30	

Set RTS

Description	This command sets the state of the RTS bit. Since RFCOMM acts as a "null modem" where RTS and CTS are connected, the remote device will see this as a change of the state of the CTS signal. In case the remote device is a RBT-001, this remote RBT-001 will send a "SPP Port Status Changed Indicator" to its application.	
PacketType	REQ	
Opcode	SPP_PORT_SET_RTS	
DataLength	2	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	State	0: False. 1: True.

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_PORT_SET_RTS	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_ILLEGAL_STATE_VALUE ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Set BREAK

Description	This command indicates that the host has detected a UART BREAK over the virtual serial link. In case the remote device is a RBT-001, this remote RBT-001 will send a "SPP Port Status Changed Indicator" to its application.	
PacketType	REQ	
Opcode	SPP_PORT_BREAK	
DataLength	2	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30.
	BreakLength 2 bytes	The length of the break in ms. Range 0 - 3000. The break length is sent in 200 ms units on the air interface (stored in a 4 bit parameter).

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_PORT_BREAK	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Set Overrun Error

Description	This command is used to indicate that the host has detected an overrun error. In case the remote device is a RBT-001, this remote RBT-001 will send a "SPP Port Status Changed Indicator" to its application.	
PacketType	REQ	
Opcode	SPP_PORT_OVERRUN_ERROR	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_PORT_OVERRUN_ERROR	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Set Parity Error

Description	This command is used to indicate that the host has detected a parity error. In case the remote device is a RBT-001, this remote RBT-001 will send a "SPP Port Status Changed Indicator" to its application.	
PacketType	REQ	
Opcode	SPP_PORT_PARITY_ERROR	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_PORT_PARITY_ERROR	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Set Framing Error

Description	This command is used to indicate that the host has detected a framing error. In case the remote device is a RBT-001, this remote RBT-001 will send a "SPP Port Status Changed Indicator" to its application.	
PacketType	REQ	
Opcode	SPP_PORT_FRAMING_ERROR	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_PORT_FRAMING_ERROR	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

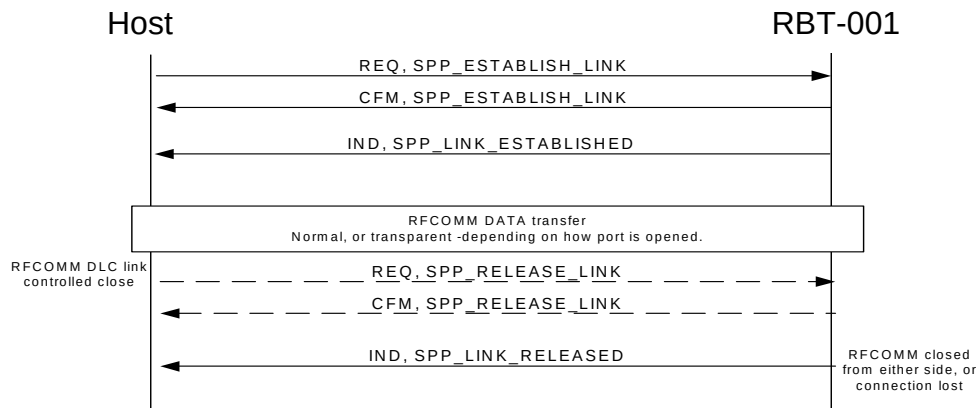
Port Status Changed

Description	Send from the RBT-001 when remote device has changed the port.			
PacketType	IND			
Opcode	SPP_PORT_STATUS_CHANGED			
DataLength	4			
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30		
	PortStatus 1 byte	Bitfield: 0000000X	DTR	0 = Low 1 = High
		Bitfield: 000000X0	RTS	0 = Low 1 = High
		Bitfield: 00000X00	DSR	0 = Low 1 = High
		Bitfield: 0000X000	CTS	0 = Low 1 = High
		Bitfield: 000X0000	Overrun Error	0 = No Error 1 = Overrun Error
		Bitfield: 00X00000	Parity Error	0 = No Error 1 = Parity Error
		Bitfield: 0X000000	Framing Error	0 = No Error 1 = Framing Error
		Bitfield:X0000000	DLC established	0 = No DLC 1 = DLC is available
	BreakLength 2 bytes	The length in ms of a break detected by the remote device. Range 0 - 3000. The value 0 is used to indicate that no break has been detected. The break length is sent in 200 ms units on the air interface.		

Establish / Release SPP link to a remote device

This section describes the basic functionality of creating a full SPP link to a remote device. Basically only one single command is needed to create the connection. The command “Establish Link” requires the BD_Addr and the RFComm port of the remote device, determined out of the Inquiry and the SDAP connection. The command and event flow can be found within the following flowchart.

The command will first be confirmed by a standard confirmation package. Afterwards the RBT-001 will start to page and try to connect to the remote device. The SPP_Link_Establishment indicator returns an error code reporting the success of the link establishment.



Establish Link

Description	Establish a DLC link to remote Bluetooth device	
PacketType	REQ	
Opcode	SPP_ESTABLISH_LINK	
DataLength	8	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	BdAddr 6 byte	Bluetooth device address for the remote device
	RemotePortNumber 1 byte	Remote device RFCOMM port number. (Must be found using SDAP)

Description	Confirm that the DLC link establishment is initiated. Note: This confirm does NOT indicate link establishment but only that link establishment is in progress. When link establishment response is received from the core, a SPP_LINK_ESTABLISHED indication is sent from the RBT-001.	
PacketType	CFM	
Opcode	SPP_ESTABLISH_LINK	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_SPP_PORT_BUSY ERROR_SPP_PORT_NOT_OPEN ERROR_SPP_INVALID_PORT ERROR_SPP_AUTOMATIC_CONNECTIONS_PROGRESSING
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Link Established

Description	Indication of establishment of a locally requested DLC link.	
PacketType	IND	
Opcode	SPP_LINK_ESTABLISHED	
DataLength	9	
Data	Status 1 byte	Refer to RFCOMM Error Codes section.
	BdAddr 6 byte	Bluetooth device address for the remote device.
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	RemotePortNumber 1 byte	Remote device RFCOMM port number.

Incoming Link Established

In case a remote device creates a link to the RBT-001, the device will indicate the successful link establishment by sending the “Incoming Link Established Indicator”. The packet includes the BD_Addr of the remote device and the local RFCOMM Port it is connected to.

Description	Indication of establishment of a remotely requested DLC link.	
PacketType	IND	
Opcode	SPP_INCOMING_LINK_ESTABLISHED	
DataLength	7	
Data	BdAddr 6 byte	Bluetooth device address for the remote device
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Release Link

Description	Release a DLC link to remote Bluetooth device	
PacketType	REQ	
Opcode	SPP_RELEASE_LINK	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

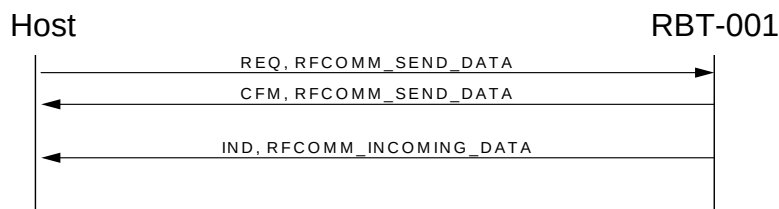
Description	Confirm that the release is initiated. When the release is complete, a SPP_LINK_RELEASED indication is sent from the RBT-001.	
PacketType	CFM	
Opcode	SPP_RELEASE_LINK	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_NO_CONNECTION ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Link Released

Description	Indicates that a DLC link is released. The link release may have been initiated locally or remote, or could be caused by a loss of link (disturbance, dead device, ...)	
PacketType	IND	
Opcode	SPP_LINK_RELEASED	
DataLength	1	
Data	Reason 1 byte	Refer to section RFCOMM Release Reasons on page 76.
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Sending / Receiving data on a SPP link (Normal mode)

In the normal mode, data are transferred in packets with normal framing.



Send Data

If not switched to transparent, data have to be sent to a remote device using this command. The local RFCOMM Port is used to address the remote device.

Description	Send data on a SPP link to remote Bluetooth device	
PacketType	REQ	
Opcode	SPP_SEND_DATA	
DataLength	3 + <PayloadSize>	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	PayloadSize 2 bytes	Number of data bytes to send. Valid range is 1 to 330 bytes..
	PayloadData <PayloadSize> bytes	The data to send.

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_SEND_DATA	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_LIMIT ERROR_UNABLE_TO_SEND ERROR_CURRENTLY_NO_BUFFER ERROR_NO_CONNECTION ERROR_SPP_INVALID_PORT ERROR_SPP_PORT_NOT_OPEN
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Incoming Data

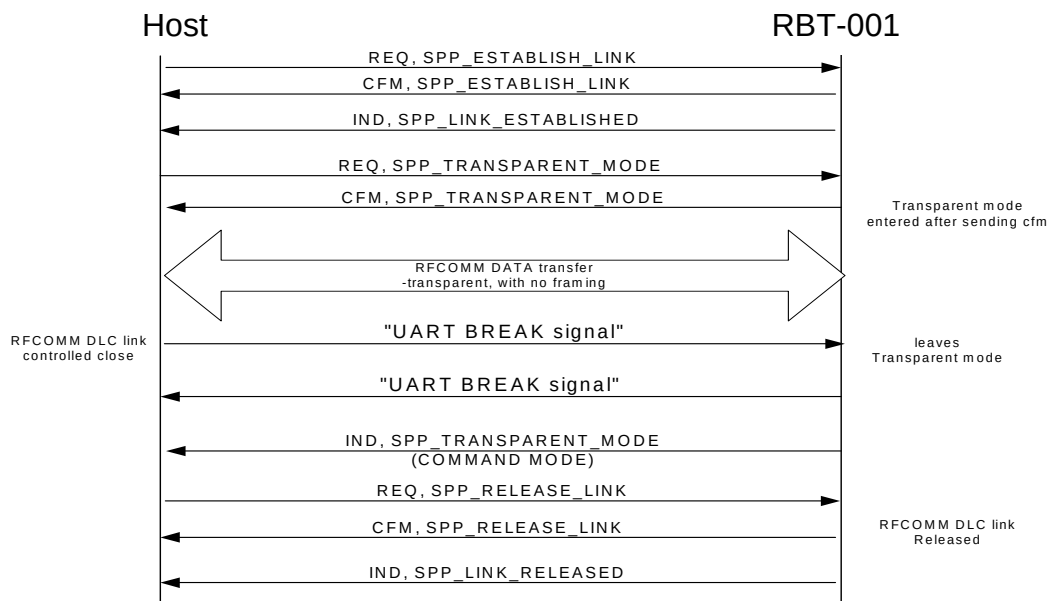
Description	Incoming data on a DLC link, from a remote Bluetooth device	
PacketType	IND	
Opcode	SPP_INCOMING_DATA	
DataLength	3 + <PayloadSize>	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	PayloadSize 2 bytes	Number of data bytes to send. Valid range is 1 to 330 bytes..
	PayloadData <PayloadSize> bytes	The data to send.

Sending / Receiving data on a SPP link (Transparent mode)

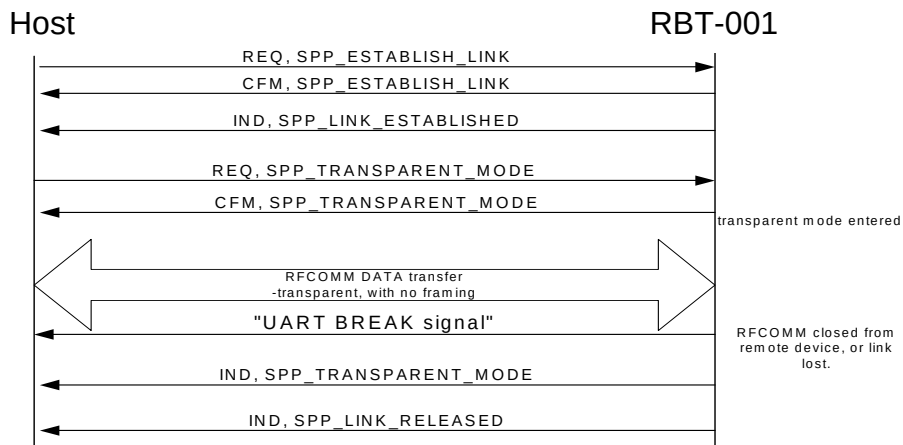
For connections with high data rate requirement, a transparent mode can be used for data transfer.

When using transparent mode data are sent / can be sent, full duplex, with no framing.

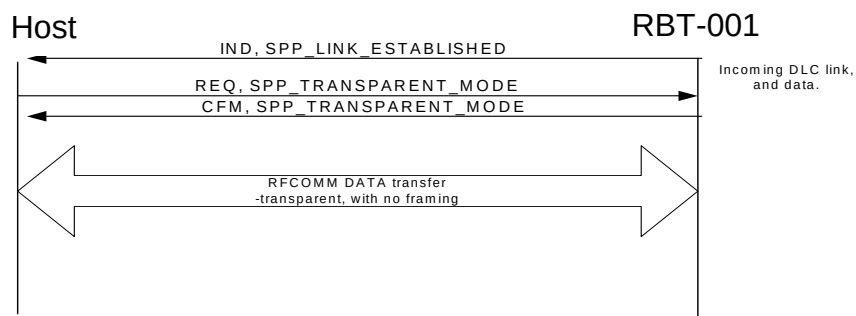
The SPP_TRANSPARENT_MODE request must be sent to switch to transparent mode. To stop the data transfer, a UART BREAK signal must be sent from the application, the RBT-001 will then leave transparent mode.



If the link is lost, the RBT-001 will issue a BREAK signal, and send the SPP_LINK_RELEASED message.



For SPP links established from remote device, a SPP_LINK_ESTABLISHED indication will be sent to the application upon reception of the first data. Again, the SPP_TRANSPARENT_MODE request can be used to switch to transparent mode.



Transparent Mode

Description	Switch to transparent mode on a SPP link to remote Bluetooth device	
PacketType	REQ	
Opcode	SPP_TRANSPARENT_MODE	
DataLength	1	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_TRANSPARENT_MODE	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_SPP_PORT_NOT_OPEN ERROR_SPP_INVALID_PORT ERROR_SPP_MULTIPLE_CONNECTIONS ERROR_NO_CONNECTION
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30

Description	Indication from RBT-001 that transparent mode is left.	
PacketType	IND	
Opcode	SPP_TRANSPARENT_MODE	
DataLength	2	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	Mode 1 byte	Command Mode = 0 Transparent Mode = 1

Default Connection

Connect Default Connection

Description	Connects the RBT-001 to a stored connection. Either a single or all connections can be established	
PacketType	REQ	
Opcode	SPP_CONNECT_DEFAULT_CON	
DataLength	1	
Data	Index 1 byte	Index in the default connection storage. Range 0x00 to 0x06. If index is set to 0xFF, all default connections will be established.

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_CONNECT_DEFAULT_CON	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_SPP_AUTOMATIC_CONNECTIONS_PROGRESSING ERROR_LIMIT ERROR_SPP_DEFAULT_CONNECTION_NOT_STORED

Store Default Connection

Description	Stores a default connection in NVS.	
PacketType	REQ	
Opcode	SPP_STORE_DEFAULT_CON	
DataLength	10	
Data	Index 1 byte	Index in the default connection storage. Range 0x00 to 0x06.
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	RemotePort 1 byte	RFCOMM port number on remote device
	RemoteBdAddress 6 bytes	The BdAddress of the remote device
	TransparentMode 1 byte	Which transparent mode to enter when the connection is established. 0x00 Transparent mode off 0x01 Transparent mode on (only for point-to-point). If multipoint connections exist the RBT-001 will not enter transparent mode

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_STORE_DEFAULT_CON	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_LIMIT ERROR_SPP_INVALID_PORT ERROR_SPP_MULTIPLE_TRANSPARENT ERROR_SPP_PORT_BUSY

Get List of Default Connections

Description	Request a list of the default connections stored in NVS	
PacketType	REQ	
Opcode	SPP_GET_LIST_DEFAULT_CON	
DataLength	1	
Data	Index 1 byte	Index in the default connection storage. Range 0x00 to 0x06.

Description	Confirm to the request above.		
PacketType	CFM		
Opcode	SPP_GET_LIST_DEFAULT_CON		
DataLength	11		
Data	Status 1 byte	ERROR_OK ERROR_LIMIT	
	Index 1 byte	Index in the default connection storage. Range 0x00 to 0x06.	
	EntryStatus 1 byte	Bitfield: 0000000x	0: Entry is not stored. 1: Entry is stored.
		Bitfield: 000000x0	0: Command mode. 1: Transparent mode.
		Bitfield: xxxxxx00	Reserved.
	RemoteBdAddress 6 bytes	The BdAddress of the remote device	
	LocalPort 1 byte	Local RFCOMM port number. Range 1-30	
	RemotePort 1 byte	RFCOMM port number on remote device	

Delete Default Connection

Description	Deletes a stored default connection in NVS	
PacketType	REQ	
Opcode	SPP_DELETE_DEFAULT_CON	
DataLength	1	
Data	Index 1 byte	Index in the default connection storage. Range 0x00 to 0x06.

Description	Confirm to the request above.	
PacketType	CFM	
Opcode	SPP_DELETE_DEFAULT_CON	
DataLength	1	
Data	Index 1 byte	Index in the default connection storage. Range 0x00 to 0x06.
	Status 1 byte	0x00 OK

Link supervision Timeout

Set Link Timeout

Description	This command is used to change the current ACL link supervision timeout. This command will affect all DLC's established to the specified device.	
PacketType	REQ	
Opcode	SPP_SET_LINK_TIMEOUT	
DataLength	8	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of the ACL link for which the Link Supervision Timeout is changed.
	LinkTimeout 2 byte	The link supervision timeout in slots (0,625ms). 0x0000: No link supervision timeout (the timer is disabled) 0x0190-0xFFFF: Valid timeout range (in slots) 0x7D00: The default value (20s)

Description	Response to the request above.	
PacketType	CFM	
Opcode	SPP_SET_LINK_TIMEOUT	
DataLength	7	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION ERROR_ILLEGAL_LINK_TIMEOUT
	BdAddr 6 bytes	The Bluetooth address of the remote device of the ACL link for which the Link Supervision Timeout is changed.

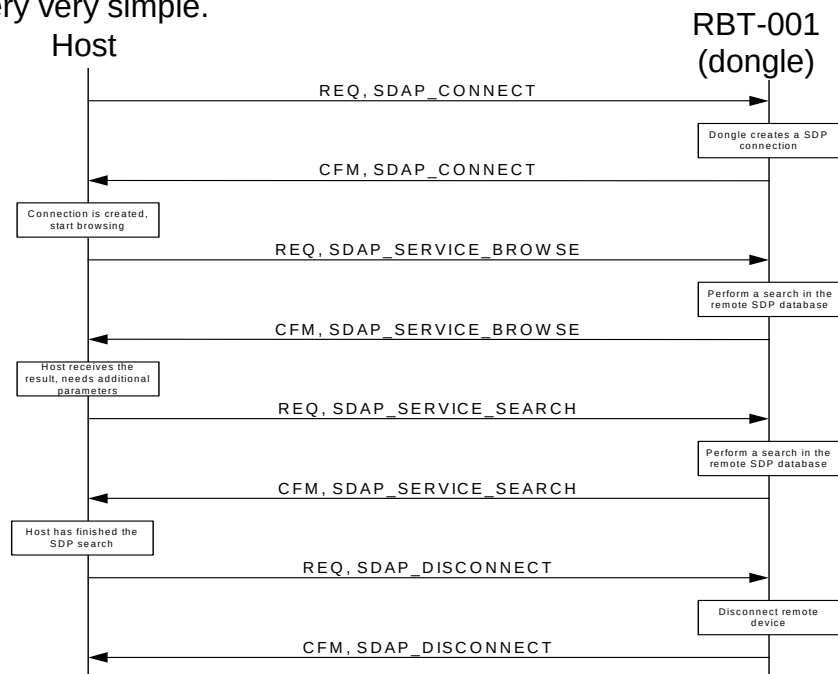
Get Link Timeout

Description	This command is used to get the current link supervision timeout setting for the given ACL link.	
PacketType	REQ	
Opcode	SPP_GET_LINK_TIMEOUT	
DataLength	6	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device of the ACL link for which the Link Supervision Timeout is requested.

Description	Response to the request above.	
PacketType	CFM	
Opcode	SPP_GET_LINK_TIMEOUT	
DataLength	9	
Data	Status 1 byte	ERROR_OK ERROR_UNSPECIFIED_ERROR ERROR_INVALID_NO_OF_PARAMETERS ERROR_NO_CONNECTION
	BdAddr 6 bytes	The Bluetooth address of the remote device of the ACL link for which the Link Supervision Timeout is requested.
	LinkTimeout 2 byte	The link supervision timeout in slots (0,625ms). 0x0000: No link supervision timeout (the timer is disabled) 0x0190-0xFFFF: Valid timeout range (in slots)

Service discovery application profile (SDAP)

The RBT-001 implements the service discovery application profile (SDAP), and makes service discovery very simple.



SDAP Connection Handling

Connect

Description	Creates an SDP connection to a remote device. Only one SDP connection can be active.	
PacketType	REQ	
Opcode	SDAP_CONNECT	
DataLength	6	
Data	BdAddr 6 bytes	The Bluetooth address of the remote device to connect to.

Description	Confirms the request above	
PacketType	CFM	
Opcode	SDAP_CONNECT	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_CONNECTION_FAILED

Disconnect

Description	This command disconnects the active SDP connection.	
PacketType	REQ	
Opcode	SDAP_DISCONNECT	
DataLength	0	

Description	Confirms the request above	
PacketType	CFM	
Opcode	SDAP_DISCONNECT	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS

Connection Lost

Description	Notification sent to the application when a loss of the SDP connection is detected.	
PacketType	IND	
Opcode	SDAP_CONNECTION_LOST	
DataLength	0	

SDAP Service Discovery

The SDAP Service Browse can be used to get the RFCOMM Port Numbers of all or only specific Service Classes. The search mechanism is based on the 16bit-UUID for the services. The actual list of UUIDs can be found within the “Bluetooth Assigned Numbers” Document of the Bluetooth SIG.

Service Browse

Description	This command is used to browse the service record of the remote device.	
PacketType	REQ	
Opcode	SDAP_SERVICE_BROWSE	
DataLength	2	
Data	BrowseGroupID 2 bytes	The requested browse group (16 bit UUID). The UUID has to be byte swapped within the command.

Description	Confirms the request above	
PacketType	CFM	
Opcode	SDAP_SERVICE_BROWSE	
DataLength	2+NoOfServices*(5+ NameLength)	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_RESULT_TOO_LARGE ERROR_TRUNCATED_ANSWER
	NoOfServices 1 byte	Number of services found on remote device.
	For each service	
	BrowseGroupID 2 bytes	The browse group UUID that the service belongs to.
	ServiceID 2 bytes	The service UUID.
	PortNr 1 byte	RFCOMM port number.
	NameLength 1 byte	The number of bytes in the service name
	ServiceName NameLength bytes	The name of the service.

Service Search

The SDAP Service Search command offers the ability to search for specific attributes for a service. The attribute IDs can be found at the Bluetooth SIG website under “Bluetooth Assigned numbers”.

NOTE: As an exception to all other commands, this command does NOT required byte swapped parameters. So the UUIDs have to be filled in as defined in "Bluetooth SIG: Bluetooth Assigned Numbers,
https://www.bluetooth.org/foundry/assignnumb/document/assigned_numbers".

Description	This command is used to search for services in the service record of the remote device.	
PacketType	REQ	
Opcode	SDAP_SERVICE_SEARCH	
DataLength	2 + 2*SearchPatternLength + 2*AttributesLength	
Data	SearchPatternLength 1 bytes	Number of 16-bit UUID's in Search pattern List. Note: Must be less than 86 elements.
	SearchPattern	List of the requested services. The search pattern list is a list of 16-bit UUID's of the requested services.
	AttributesLength 1 byte	Number of 16-bit UUID's in attributeId list. Note: Must be less than 86 elements.
	Attributes	List of requested attributes for the requested services. The attribute list is a list of 16-bit UUID's for the requested attributes.

Description	Confirms the request above	
PacketType	CFM	
Opcode	SDAP_SERVICE_SEARCH	
DataLength	3 + Lenght	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_UNKNOWN_ERROR ERROR_RESULT_TOO_LARGE
	Length 2 byte	Length of the result of the search. Maximum 330 bytes.
	Result Length bytes	Result of the search

Service Request

Each Service Entry has a unique number called “Service Record Handle”. This command is used to get the record handle for stored entries for specific UUIDs.

Description	This command is used the service record handles, from a remote device, for the given services in the search pattern.	
PacketType	REQ	
Opcode	SDAP_SERVICE_REQUEST	
DataLength	2 + 2* SearchPatternLength	
Data	SearchPatternLength 1 bytes	Number of 16-bit UUID’s in Search pattern List. Note: Must be less than 86 elements.
	SearchPattern <2*SearchPatternLength > bytes	List of the requested services. The search pattern list is a list of 16-bit UUID’s of the requested services

Description	Confirms the request above	
PacketType	CFM	
Opcode	SDAP_SERVICE_REQUEST	
DataLength	2 + 4*Lenght	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_UNKNOWN_ERROR ERROR_RESULT_TOO_LARGE
	Length 1 byte	Number of 32-bit service record handles returned from remote device.
	Result <4*Length> bytes	The received 32-bit service record handles from the remote device.

Attribute Request

Instead of browsing the whole list of services within the remote database it is also possible to search only for specific attributes within a chosen entry. This command is based on the attribute “connection handle” of that specific entry.

Connection handles can be retrieved by the “SDAP Service Search” Command.

Description	This command is used to get the given attributes for a given service record handle.	
PacketType	REQ	
Opcode	SDAP_ATTRIBUTE_REQUEST	
DataLength	5 + 2*AttributesLength	
Data	Handle 4 bytes	The 32-bit service record handle returned for a given service by SDAP_ATTRIBUTE_REQUEST.
	AttributesLength 1 byte	Number of 16-bit UUID's in attributeId list. Note: Must be less than 86 elements.
	Attributes <2*AttributesLength > bytes	List of requested attributes for the requested services. The attribute list is a list of 16-bit UUID's for the requested attributes.

Description	Confirms the request above	
PacketType	CFM	
Opcode	SDAP_ATTRIBUTE_REQUEST	
DataLength	3 + Length	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_UNKNOWN_ERROR ERROR_RESULT_TOO_LARGE
	Length 2 byte	Length of the result of the search. Maximum 330 bytes.
	Result	Result of the search

Configuration

Change NVS UART Speed

Description	This command will change the UART speed stored in NVS. The new UART speed will be used after a reset.	
PacketType	REQ	
Opcode	CHANGE_NVS_UART_SPEED	
DataLength	1	
Data	UartSpeed 1 Byte	The UART speed to be stored in NVS 0x00 = 2400 0x01 = 4800 0x02 = 7200 0x03 = 9600 0x04 = 19200 0x05 = 38400 0x06 = 57600 0x07 = 115200 0x08 = 230400 0x09 = 460800 0x0A = 921600

Description	Confirms the request above	
PacketType	CFM	
Opcode	CHANGE_NVS_UART_SPEED	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_UART_SPEED_OUT_OF_RANGE ERROR_INVALID_NO_OF_PARAMETERS

Change UART Settings

Description	This command will change the UART settings stored in NVS. The new UART settings will be used after a reset.	
PacketType	REQ	
Opcode	CHANGE_UART_SETTINGS	
DataLength	2	
Data	PartityBit 1 Byte	0x00 None 0x01 Even 0x02 Odd
	StopBits 1 Byte	0x00 One Stop bit 0x01 Two Stop bits

Description	Confirms the request above	
PacketType	CFM	
Opcode	CHANGE_UART_SETTINGS	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_PARITY_BIT_OUT_OF_RANGE, ERROR_STOP_BITS_OUT_OF_RANGE, ERROR_INVALID_NO_OF_PARAMETERS

Read Operation Mode

Description	This command will read out the current operation mode.	
PacketType	REQ	
Opcode	READ_OPERATION_MODE	
DataLength	0	
Data	None	

Description	Confirms the request above	
PacketType	CFM	
Opcode	READ_OPERATION_MODE	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	Mode 1 Byte	0x00 Automatic Operation OFF 0x01 Automatic Operation ON

Write Operation Mode

Description	This command will change the operation mode stored in NVS. The new setting will take effect after a reset.	
PacketType	REQ	
Opcode	WRITE_OPERATION_MODE	
DataLength	1	
Data	Mode 1 Byte	0x00 Automatic Operation OFF 0x01 Automatic Operation ON

Description	Confirms the request above	
PacketType	CFM	
Opcode	READ_OPERATION_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_INVALID_MODE

Set Ports To Open

Description	This command will change which RFCOMM ports the RBT-001 will open both at start-up and runtime. The value gets effective immediately and is stored within the NVS to be restored after reset.	
PacketType	REQ	
Opcode	SET_PORTS_TO_OPEN	
DataLength	4	
Data	PORTS 4 Bytes	This field is a 32-bit mask indicating which RFCOMM ports the RBT-001 has to open. Bit 30 and 31 must be set to 0. Bit 0 is RFCOMM port 1 and bit 29 is port 30 e.g. if this field has the value 0x00000007, port 1 to 3 will be opened. All other ports will be closed if open.

Description	Confirms the request above	
PacketType	CFM	
Opcode	PORTS_TO_OPEN	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_PORT ERROR_INVALID_NO_OF_PARAMETERS

Get Ports To Open

Description	This command will get the value of which RFCOMM ports the RBT-001 will open both at start-up and runtime.	
PacketType	REQ	
Opcode	GET_PORTS_TO_OPEN	
DataLength	0	
Data		

Description	Confirms the request above	
PacketType	CFM	
Opcode	PORTS_TO_OPEN	
DataLength	5	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	PORTS 4 Bytes	This field is a 32-bit mask indicating which RFCOMM ports the RBT-001 has to open. Bit 30 and 31 must be set to 0. Bit 0 is RFCOMM port 1 and bit 29 is port 30 e.g. if this field has the value 0x00000007, port 1 to 3 will be opened.

Restore Factory Settings

Description	This command will restore the RBT-001 configuration in NVS to factory settings. The factory settings are similar to all default settings listed in the table on page 13. The RBT-001 needs to be restarted in order to let the changes take effect. No commands that write to NVS should be sent to the device in between RESTORE_FACTORY_SETTINGS REQ and CFM. When CFM is sent all the factory settings are completed.	
PacketType	REQ	
Opcode	RESTORE_FACTORY_SETTINGS	
DataLength	0	
Data	none	

Description	Confirms the request above	
PacketType	CFM	
Opcode	RESTORE_FACTORY_SETTINGS	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS

Store Class of Device

Description	This command will store the class of device for the RBT-001 in NVS. The proper value for the class of device parameter is specified by the Bluetooth SIG.	
PacketType	REQ	
Opcode	STORE_CLASS_OF_DEVICE	
DataLength	3	
Data	ClassOfDevice 3 bytes	The class of device value to be stored in NVS.

Description	Confirms the request above	
PacketType	CFM	
Opcode	STORE_CLASS_OF_DEVICE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS

Force Master Role

The Force Master Role is initiated only after boot up or reset. The role is stored as parameter within NVS. If Force Master is activated the RBT-001 tries to switch its role to master if connected from another device. For this the RBT-001 is sending a Master/Slave switch request after link establishment to the remote device. If the switch is successful, the link setup will be continued and the RBT-001 is open for other incoming connections. If the switch fails, the link will be dropped.

This functionality allows Access Point like applications on the RBT-001.

Description	This command will change the preferred role of the RBT-001 e.g. to force master role at any connections. The RBT-001 must be reset to let the changes take effect.	
PacketType	REQ	
Opcode	FORCE_MASTER_ROLE	
DataLength	1	
Data	Role 1 bytes	0x00 Don't care about role 0x01 Force master role at connection setup

Description	Confirms the request above	
PacketType	CFM	
Opcode	FORCE_MASTER_ROLE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_ROLE ERROR_INVALID_NO_OF_PARAMETERS

Set Default Link Policy

Description	This command is used to change the default link policy. The default link policy is set during connection setup. The default link policy setting is stored in NVS.	
PacketType	REQ	
Opcode	SET_DEFAULT_LINK_POLICY	
DataLength	2	
Data	LinkPolicy 2 byte	Bitfield: 0x0001 = Master-slave switch allowed 0x0002 = Hold mode allowed 0x0004 = Sniff mode allowed 0x0008 = Park mode allowed

Description	Response to the request above.	
PacketType	CFM	
Opcode	SET_DEFAULT_LINK_POLICY	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_ILLEGAL_LINK_POLICY

Get Default Link Policy

Description	This command is used to read the default link policy setting from NVS.	
PacketType	REQ	
Opcode	GET_DEFAULT_LINK_POLICY	
DataLength	0	

Description	Response to the request above.	
PacketType	CFM	
Opcode	GET_DEFAULT_LINK_POLICY	
DataLength	2	
Data	LinkPolicy 2 byte	Bitfield: 0x0001 = Master-slave switch allowed 0x0002 = Hold mode allowed 0x0004 = Sniff mode allowed 0x0008 = Park mode allowed

Set Event Filter

Description	This command is used to set the event filter. The setting is stored in NVS.	
PacketType	REQ	
Opcode	SET_EVENT_FILTER	
DataLength	2	
Data	Filter 1 byte	0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported, UART break still generated and detected. 0x03: No events generated, UART break not generated or detected (full cable replacement)

Description	Response to the request above.	
PacketType	CFM	
Opcode	SET_EVENT_FILTER	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_LIMIT

Get Event Filter

Description	This command reads the current event filter setting.	
PacketType	REQ	
Opcode	GET_EVENT_FILTER	
DataLength	0	

Description	Response to the request above.	
PacketType	CFM	
Opcode	GET_EVENT_FILTER	
DataLength	1	
Data	Filter 1 byte	0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported, UART break still generated and detected. 0x03: No events generated, UART break not generated or detected (full cable replacement)

Set Default Link Timeout

Description	This command is used to change the default link supervision timeout. The default link supervision timeout is set during connection setup. The default link supervision timeout setting is stored in NVS.	
PacketType	REQ	
Opcode	SET_DEFAULT_LINK_TIMEOUT	
DataLength	2	
Data	LinkTimeout 2 byte	The link supervision timeout in slots (0,625ms). The default value stored in the NVS after a factory reset is 20s. 0x0000: No link supervision timeout (the timer is disabled) 0x0190-0xFFFF: Valid timeout range (in slots) 0x7D00: The default value (20s)

Description	Response to the request above.	
PacketType	CFM	
Opcode	SET_DEFAULT_LINK_TIMEOUT	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_ILLEGAL_LINK_TIMEOUT

Get Default Link Timeout

Description	This command is used to read the default link supervision timeout setting from NVS.	
PacketType	REQ	
Opcode	GET_DEFAULT_LINK_TIMEOUT	
DataLength	0	

Description	Response to the request above.	
PacketType	CFM	
Opcode	GET_DEFAULT_LINK_TIMEOUT	
DataLength	2	
Data	LinkTimeout 2 byte	The link supervision timeout in slots. 0x0000: No link supervision timeout (the timer is disabled) 0x0190-0xFFFF: Valid timeout range (in slots)

Set Default Link Latency

Description	This command is used to change the default SPP link latency. The default link latency is set during SPP connection setup. The default link latency setting is stored in NVS. The link latency is used to calculate a poll interval for the ACL link. The calculated poll interval may be bigger than the latency specified because of e.g. another ACL link.	
PacketType	REQ	
Opcode	SET_DEFAULT_LINK_LATENCY	
DataLength	2	
Data	Latency 2 byte	The link latency in slots. 0x0000: No link latency requirement (default) 0x0002 - 0x0190: Valid link latency In case 0x0000 is set the link latency is managed by the master, typically 40 slots (0x0028, 25ms) in a point-to-point connection are used

Description	Response to the request above.	
PacketType	CFM	
Opcode	SET_DEFAULT_LINK_LATENCY	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS ERROR_LIMIT

Get Default Link Latency

Description	This command is used to read the default link latency setting from NVS.	
PacketType	REQ	
Opcode	GET_DEFAULT_LINK_LATENCY	
DataLength	0	

Description	Response to the request above.	
PacketType	CFM	
Opcode	GET_DEFAULT_LINK_LATENCY	
DataLength	2	
Data	Latency 2 byte	The link latency in slots. 0x0000: No link latency requirement (default) 0x0002 - 0x0190: Valid link latency In case 0x0000 is set the link latency is managed by the master, typically 40 slots (0x0028, 25ms) in a point-to-point connection are used

SDP Record Handling

The RBT-001 allows the modification of the Local Service Discovery Database. By default, the service database contains one entry configured at RFComm port 1 for a Serial Port Profile, with Authentication and Encryption enabled.

If the application needs to offer a second service to another device, this service has to be entered within the service database and, if required, the RFComm port has to be opened. The service entry in general includes information about the name of the service, which appears on a remote device after browsing, the port number and security settings.

Each registered service entry itself can be enabled or disabled. This allows to have different services registered to one specific RFCOMM port enabling and disabling them by needs.

The command set includes just one command to store any SDP record available within the Service Database, called "Store Generic SDP Record". Each record includes the profile specific settings needed.

The service records are stored in a database like system within the NVS, so they are still available after reset. The storage of an entry is confirmed by the RBT-001 with a specific record identifier. This identifier is needed for the Enabling or Disabling command to address those specific entries.

Enable SDP Record

Description	This command is used to enable/disable stored SDP records in the RBT-001.	
PacketType	REQ	
Opcode	ENABLE_SDP_RECORD	
DataLength	2	
Data	State 1 byte	The new state of the SDP record. 0x00 Disable the record. 0x01 Enable the record.
	Identifier 1 byte	The identifier received when the record was stored in the RBT-001.

Description	Confirms the request above	
PacketType	CFM	
Opcode	ENABLE_SDP_RECORD	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_ILLEGAL_STATE_VALUE ERROR_IDENTIFIER_OUT_OF_RANGE ERROR_RECORD_ALREADY_IN_SELECTED_STATE ERROR_IDENTIFIER_NOT_IN_USE ERROR_INVALID_NO_OF_PARAMETERS
	Identifier 1 byte	The identifier received when the record was stored in the RBT-001.

Delete All SDP Records

Description	This command is used to delete all stored SDP records in the RBT-001.	
PacketType	REQ	
Opcode	DELETE_SDP_RECORDS	
DataLength	0	
Data	None	

Description	Confirms the request above	
PacketType	CFM	
Opcode	DELETE_SDP_RECORDS	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS

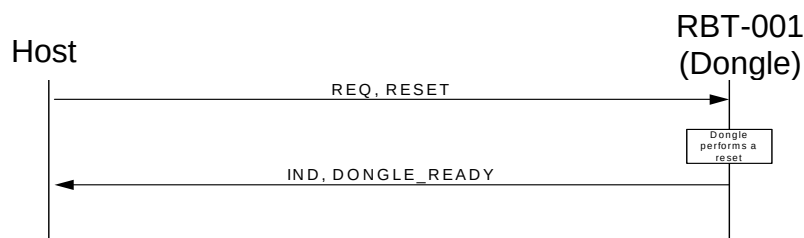
Store Generic SDP Record

Description	This command will create a new service record in the local SDP database, stored in the NVS.	
PacketType	REQ	
Opcode	STORE_SDP_RECORD	
DataLength	5 + <SdpRecordLength>	
Data	LocalPort 1 byte	The local RFCOMM port used by this service. Range 1 to 30.
	Authentication 1 byte	0x00 No authentication requirements. 0x02 Authentication is only required for this profile for incoming connections. 0x20 Authentication is only required for this profile for outgoing connections. 0x22 Authentication is required for this profile for connections in both directions
	Encryption 1 byte	0x00 No encryption requirements 0x04 Encryption is only required for this profile for incoming connections. 0x40 Encryption is only required for this profile for outgoing connections. 0x44 Encryption is required for this profile for connections in both directions
	SdpRecordLength 2 bytes	The number of bytes in the SDP record.
	SdpRecord <SdpRecord- Length> Bytes	The SDP record data formatted as: < uint16:AttributeId, SdpDataElement:AttributeData, uint16:AttributeId, SdpDataElement:AttributeData, ...>. The maximum length for each service is 255 bytes.

Description	Confirms the request above	
PacketType	CFM	
Opcode	STORE_SDP_RECORD	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_SDP_RECORD ERROR_MAXIMUM_NO_OF_SERVICE_RECORDS_REACHED ERROR_WRITING_TO_NVS
	Identifier 1 byte	A unique identifier, which must be used when dynamic enabling/disabling the record.

Other Commands

Reset



Description	This command will perform a soft reset of the RBT-001. The RBT-001 will send a DEVICE_READY indication when it has performed the reset.	
PacketType	REQ	
Opcode	RESET	
DataLength	0	
Data	None	

Device Ready

Description	The device will send this indication to the host when the device is fully initialized and ready to receive commands from the host.	
PacketType	IND	
Opcode	DEVICE_READY	
DataLength	1+Length	
Data	Length 1 byte	Number of bytes in software version string
	version <Length> bytes	ASCII string containing the software version. e.g. "0210" indicating that the software version is version 2.10.

Test Mode

Description	This command will activate the test mode. To exit the test mode, a reset of the device must be performed. If local loopback mode is activated all data send to the device though the UART, are send back to the host. The device can only leave local loopback mode by performing a hardware reset.	
PacketType	REQ	
Opcode	TEST_MODE	
DataLength	1	
Data	Mode 1 byte	The Bluetooth test mode to enter: 0x01 Enable Bluetooth test mode 0x02 Enable local loopback mode

Description	Confirms the request above	
PacketType	CFM	
Opcode	TEST_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_ILLEGAL_TESTMODE ERROR_UNKNOWN_ERROR ERROR_INVALID_NO_OF_PARAMETERS

RF TEST MODE

Description	This command will activate the RF test mode. To exit the test mode, a reset of the device must be performed. The transmitter test must be stopped before a new test with changed parameters can be started.	
PacketType	REQ	
Opcode	RF_TEST_MODE	
DataLength	14	
Data	Test 1 byte	Test Scenario: 0x00: Stop transmit 0x01: Burst transmit, take payload from this command 0x02: Burst transmit, use PRBS-9 sequence 0x03: Continuous transmit, sends all-1 if ModulationPayload is nonzero, otherwise all-0 0x04: Hopping receive 0x05: continuous receive at fixed channel
	Channel 1 byte	Channel number 0 – 78 0: 2402 MHz 78: 2480 MHz 255 (0xFF): Hopping in connection state

	PaCtrl 1 byte	Reserved for future use
	ModulationCtrl 1 byte	Modulation Control: 0: No modulation 1: Access code only (68 us TX data every 1250 us) 2: Bluetooth DH1 packet with defined payload 3: Bluetooth DM1 packet (17 bytes) 4: Bluetooth DH1 packet (27 bytes) 5: Invalid 6: Invalid 7: Bluetooth HV3 packet (30 bytes) 8: Invalid 9: Bluetooth AUX1 packet (29 bytes) 10: Bluetooth DM3 packet (121 bytes) 11: Bluetooth DH3 packet (183 bytes) 12: Invalid 13: Invalid 14: Bluetooth DM5 packet (224 bytes) 15: Bluetooth DH5 packet (339 bytes)
	ModulationPattern 8 bytes	Defines Access Code modulation if ModulationCtrl = 1, else ignored.
	ModulationPayload 1 byte	Defines one byte of payload repeated through packet if Test = 1, else ignored
	Options 1 byte	0: Whitening is disabled for transmit. 1: Whitening is enabled for transmit. 2-255: Reserved for future use.

Description	Confirms the request above	
PacketType	CFM	
Opcode	RF_TEST_MODE	
DataLength	1	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS

Read RSSI

Description	This command will read out the current RSSI value for given link a remote device.	
PacketType	REQ	
Opcode	READ_RSSI	
DataLength	6	
Data	BdAddress 6 byte	Bluetooth Address of the remote device for which to read out the current RSSI value.

Description	Confirms the request above	
PacketType	CFM	
Opcode	READ_RSSI	
DataLength	2	
Data	Status 1 byte	ERROR_OK ERROR_UNKNOWN_ERROR ERROR_INVALID_NO_OF_PARAMETERS
	RSSI 1 byte	The RSSI value Range: 0x01: Too High 0x00: Good 0xFF: Too Low

Read NVS

Description	This command can be used to read from the eeprom (NVS) in the RBT-001 module.	
PacketType	REQ	
Opcode	READ_NVS	
DataLength	3	
Data	Address 2 bytes	The address.
	Length 1 byte	The number of bytes that should be read.

Description	Confirms the request above.	
PacketType	CFM	
Opcode	READ_NVS	
DataLength	4 + Length	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	Address 2 bytes	The address.
	Length 1 byte	The number of bytes that has been read.
	Data < Length > bytes	The data.

Write NVS

Description	This command can be used to write data to the NVS in the RBT-001 module.	
PacketType	REQ	
Opcode	WRITE_NVS	
DataLength	3 + Length	
Data	Address 2 bytes	The address.
	Length 1 byte	The number of bytes that should be written.
	Data < Length > bytes	The data.

Description	Confirms the request above.	
PacketType	CFM	
Opcode	WRITE_NVS	
DataLength	4	
Data	Status 1 byte	ERROR_OK ERROR_INVALID_NO_OF_PARAMETERS
	Address 2 bytes	The address.
	Length 1 byte	The number of bytes that has been written

ErrorCodes

ACL Error Codes

Error Code	Description
0x01	Unknown HCI Command.
0x02	No Connection.
0x03	Hardware Failure.
0x04	Page Timeout.
0x05	Authentication Failure.
0x06	Key Missing.
0x07	Memory Full.
0x08	Connection Timeout.
0x09	Max Number Of Connections.
0x0A	Max Number Of SCO Connections To A Device.
0x0B	ACL connection already exists.
0x0C	Command Disallowed.
0x0D	Host Rejected due to limited resources.
0x0E	Host Rejected due to security reasons.
0x0F	Host Rejected due to remote device is only a personal device.
0x10	Host Timeout.
0x11	Unsupported Feature or Parameter Value.
0x12	Invalid HCI Command Parameters.
0x13	Other End Terminated Connection: User Ended Connection.
0x14	Other End Terminated Connection: Low Resources.
0x15	Other End Terminated Connection: About to Power Off.
0x16	Connection Terminated by Local Host.
0x17	Repeated Attempts.
0x18	Pairing Not Allowed.
0x19	Unknown LMP PDU.
0x1A	Unsupported Remote Feature.
0x1B	SCO Offset Rejected.
0x1C	SCO Interval Rejected.
0x1D	SCO Air Mode Rejected.
0x1E	Invalid LMP Parameters.
0x1F	Unspecified Error.
0x20	Unsupported LMP Parameter Value.
0x21	Role Change Not Allowed
0x22	LMP Response Timeout
0x23	LMP Error Transaction Collision
0x24	LMP PDU Not Allowed
0x25	Encryption Mode Not Acceptable

0x26	Unit Key Used
0x27	QoS is Not Supported
0x28	Instant Passed
0x29	Pairing with Unit Key Not Supported
0x2A	Different Transaction Collision
0x2B	Reserved
0x2C	QoS Unacceptable Parameter
0x2D	QoS Rejected
0x2E	Channel Classification Not Supported
0x2F	Insufficient Security
0x30	Parameter out of Mandatory Range
0x31	Reserved
0x32	Role Switch Pending
0x33	Reserved
0x34	Reserved Slot Violation
0x35	Role Switch Failed

Generic error codes

Error code	Macro	Description
0x00	ERROR_OK	No error.
0x01	ERROR_INVALID_NO_OF_PARAMETERS	The number of bytes in the request does not correspond to the protocol specification
0x02	ERROR_DURATION_OUT_OF_RANGE	The given duration value is not valid according to the specification.
0x03	ERROR_INVALID_MODE	The selected mode is not valid according to the specification
0x04	ERROR_TIMEOUT	A timeout occurred.
0x05	ERROR_UNKNOWN_ERROR	An unknown error occurred.
0x06	ERROR_NAME_TOO_LONG	The number of bytes in the name string is longer than the maximum specified value.
0x07	ERROR_INVALID_DISCOVERABILITY_PARAMETER	The given discoverability parameter does not contain a valid value according to the specification.
0x08	ERROR_INVALID_CONNECTABILITY_PARAMETER	The given connectability parameter does not contain a valid value according to the specification.
0x09	ERROR_INVALID_SECURITY_MODE	The given security mode is not a valid Bluetooth security mode.
0x0a	ERROR_LINKKEY_DOES_NOT_EXISTS	No link key exists for the given Bluetooth address
0x0b	ERROR_CONNECTION_FAILED	The connection setup failed due to unknown reasons.
0x0c	ERROR_TRUNCATED_ANSWER	The returned number of services is too large to be handled by the RBT-001. The answer is truncated
0x0d	ERROR_RESULT_TOO_LARGE	The SDP result from the remote device is too large to be handled by the RBT-001 due to ram limitations
0x0e	ERROR_NOT_POSSIBLE_TO_ENTER_TESTMODE	It is currently not possible to enter the selected test mode.

Error code	Macro	Description
0x0f	ERROR_ILLEGAL_TESTMODE	The given test mode is not a valid test mode according to the specification
0x10	ERROR_RESET_TO_NSC_BDADDRESS	The RBT-001 will change the Bluetooth address to the NSC address.
0x11	ERROR_UART_SPEED_OUT_OF_RANGE	The selected UART speed value is not valid according to the specification.
0x12	ERROR_INVALID_PORT	The given port value is larger than the maximum specified value.
0x13	ERROR_ILLEGAL_STATE_VALUE	The given state value is not a valid state according to the specification
0x14	ERROR_IDENTIFIER_OUT_OF_RANGE	The given identifier is larger than the maximum specified value.
0x15	ERROR_RECORD_ALREADY_IN_SELECTED_STATE	The service record is already enabled/disabled.
0x16	ERROR_INVALID_AUTHENTICATION_VALUE	The given authentication value is not a valid value according to the specification.
0x17	ERROR_INVALID_ENCRYPTION_VALUE	The given encryption value is not a valid value according to the specification.
0x18	ERROR_MAXIMUM_NO_OF_SERVICE_RECORDS_REACHED	The maximum number of service records, which the RBT-001 is able to store, is reached.
0x19	ERROR_WRITING_TO_NVS	An error occurred while writing to flash. The service record may not be stored.
0x1a	ERROR_INVALID_ROLE	The given role value is not a valid value according to the specification.
0x1b	ERROR_LIMIT	Limits exceeded (Parameter(s) violates limits).
0x1c	ERROR_UNEXPECTED	Unexpected at this moment.
0x1d	ERROR_UNABLE_TO_SEND	Could not send at this moment, no reason specified.
0x1e	ERROR_CURRENTLY_NO_BUFFER	Currently no room in buffer, try again later.
0x1f	ERROR_NO_CONNECTION	Trying to use an inexistent connection.
0x20	ERROR_SPP_INVALID_PORT	Port number out of range.
0x21	ERROR_SPP_PORT_NOT_OPEN	Port is closed.
0x22	ERROR_SPP_PORT_BUSY	Connection establishment on a PORT that has a connection.
0x23	ERROR_SPP_MULTIPLE_CONNECTIONS	Transparent mode attempted while more than 1 connection active.
0x24	ERROR_SPP_MULTIPLE_TRANSPARENT	Trying to store a default connection when a transparent default connection is already stored, or trying to store a transparent default connection when another connection is already stored.
0x25	ERROR_SPP_DEFAULT_CONNECTION_NOT_STORED	Trying to connection to a default connection, which is not stored.
0x26	ERROR_SPP_AUTOMATIC_CONNECTIONS_PROGRESSING	Trying to start connecting to default connections when default connection establishment is already progressing.
0x27	ERROR_UNSPECIFIED_ERROR	Other error.
0x28	ERROR_IDENTIFIER_NOT_IN_USE	Trying to enable a SDP record which is not stored. Wrong identifier.
0x29	ERROR_INVALID_SUPPORTED_FAXCLASS_VALUE	Faxclass parameter must be 0 or 1.
0x2a	ERROR_T00_MANY_SUPPORTED_FORMATS	The given number of supported formats exceeds the specified maximum number of supported formats.
0x2b	ERROR_T00_MANY_DATASTORES	The given number of data stores excess the specified maximum number of data stores.
0x2C	ERROR_ATTEMPT_FAILED	Attempt to change low power mode failed
0x2D	ERROR_ILLEGAL_LINK_POLICY	The given link policy value is out of range
0x2E	ERROR_PINCODE_LENGTH	The pin code length field is zero or too large.
0x2F	ERROR_PARITY_BIT_OUT_OF_RANGE	The given parity check is out of range

Error code	Macro	Description
0x30	ERROR_STOP_BITS_OUT_OF_RANGE	The given number of stop bits is out of range.
0x31	ERROR_ILLEGAL_LINK_TIMEOUT	The given link timeout value is out of range.
0x32	ERROR_COMMAND_DISALLOWED	The command is not allowed.
0x33	ERROR_ILLEGAL_AUDIO_CODEC_TYPE	The given Audio CODEC type is out of range.
0x34	ERROR_ILLEGAL_AUDIO_AIR_FORMAT	The given Audio Air format is out of range.
0x35	ERROR_SDP_RECORD_TOO_LONG	The SDP record is too long.
0x36	ERROR_SDP_FAILED_TO_CREATE_RECORD	The SDP server failed to create the SDP record.
0x37	ERROR_SET_VOLUME_FAILED	The selected codec does not support volume control.
0x38	ERROR_ILLEGAL_PACKET_TYPE	The packet type specified in the request is not valid.
0x39	ERROR_INVALID_CODEC_SETTING	The codec (slave) setting is invalid.

RFCOMM Error Codes

Error code	Macro	Description
0x00	RFCS_NO_ERROR	No error
0x01	RFCS_INVALID_DLC	The DLC does not exist
0x02	RFCS_INVALID_PORT	The port does not exist
0x03	RFCS_DLC_ESTABLISH_FAILED	The DLC establishment failed
0x04	RFCS_ACCESS_REJECTED	SECM did not authorize access to the requested service (DLC)
0x05	RFCS_INVALID_CONNECTION	There does not exist a DLC/L2CAP connection to the device
0xFF	RFCS_UNSPECIFIED_ERROR	Not used

RFCOMM Release Reasons

Error code	Macro	Description
0x00	RFCR_DLC_DISC_LOCAL_DEVICE	The local device has disconnected the DLC.
0x01	RFCR_DLC_DISC_REMOTE_DEVICE	The remote device has disconnected the DLC.
0x02	RFCR_DLC_DISC_ACL_FAILURE	ACL link failure/ link supervision timeout.
0x03	RFCR_DLC_DISC_LOWER_LAYER	Lower layer (e.g. L2CAP) has disconnected the DLC.

Bibliography

- ❑ RoboTech: RBT-001 Datasheet
- ❑ National Semiconductor: LMX9830 Software Users Guide
- ❑ Bluetooth SIG: Specification of the Bluetooth System
- ❑ Bluetooth SIG: Bluetooth Assigned Numbers,
https://www.bluetooth.org/foundry/assignnumb/document/assigned_numbers

Certifications

Bluetooth

The RBT-001 module has passed the Bluetooth Qualification/Certification process and has been fully qualified as "end product" with QPID: B0313300.

CE

The RBT-001 module complies with the requirements of the R&TTE Directive 99/05/CE:

- ☐ EN 60950-1:2001 - LV Art. 3.1a 99/05/CE
- ☐ ETSI EN 301.489-03 V1.4.1 - EMC Art. 3.1b 99/05/CE
- ☐ EN 300 328 V1.6.1, EN 300 328-2 V1.2.1 - ERM Art. 3.2 99/05/CE

Revision History

Revision # (PDF Date)	Revisions / Comments
1.0 (31-Oct-06)	Initial release
1.1 (24-Nov-06)	Changed maximum Voltage DC input Added " Powering with Voltage DC Input above 3.3V " Added " RS232 adapter board schematic "
1.2 (01-Sep-07)	Updated " Powering with Voltage DC Input above 3.0V " Added " Certifications "

AppNote: Easy Bluetooth to Easy Bluetooth Communication

By popular demand, this is an application note to explain how to connect two Easy Bluetooth modules together to communicate. In this example, one Easy Bluetooth is communicating to another Easy Bluetooth sending bytes of data as a variable "counter". This example is pretty simplistic, but can be modified to send more complex data packets. The PC that I used is running WinXP Pro, so all the instructions will be for XP; if you Vista or Win7, can refer to the user guide or control panel to configure the Bluetooth modules.



Features

- Allows wireless communication between microcontrollers
- Addressable modules to identify a Bluetooth module
- Build-in error correction within Bluetooth protocol
- Ability to communicate with a Bluetooth dongle on a PC up to ~10 meters

Application Ideas

- Multi-controller wireless communication
- Control one robot with another one without the use of a PC
- Wireless data transfers

Materials Used

Part #	Description	Quantity
30085	Easy Bluetooth Module	2
28803	BOE Full Kit (USB)	2
750-00009	7.5 V 1 A power supply	2
DBT-120	D-Link Bluetooth Dongle *	1

** Materials noted with a star (*) are not carried by Parallax Inc., and will need to be obtained through separate means. The Bluetooth dongle is used for configuration purposes.*

Assembly Instructions

Step 1: Unpack & Check

Unpack and verify all parts are accounted for.

2 x Easy Bluetooth Modules

2 x BOE Full Kit (USB)

- 2 x Board of Education
- 2 x BS2
- 2 x USB Cable
- 2 x 10 pack wire
- 2 x Rubber Feet

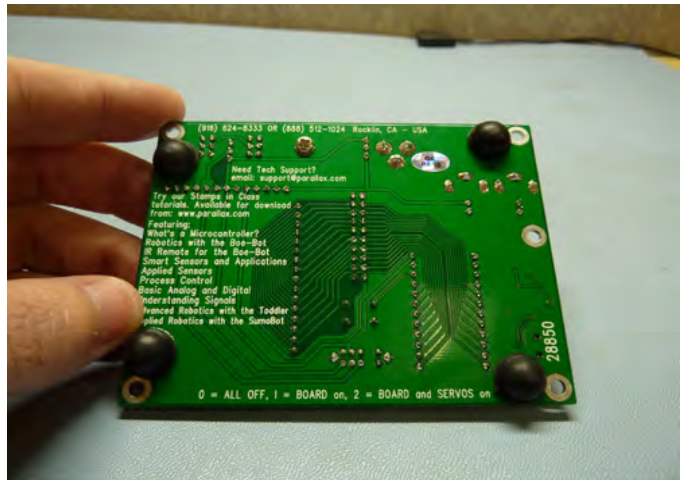
2 x 7.5VDC 1A Power Supply

** Bluetooth dongle is not included but still needed; obtain a dongle or compatible module for configuration purposes.*

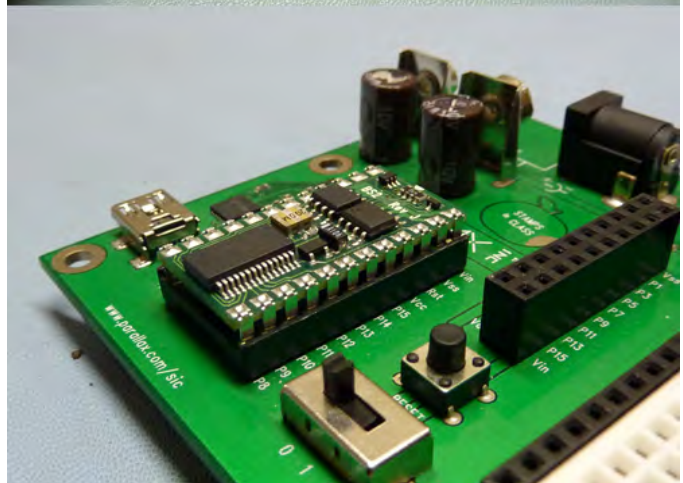


Step 2: Assemble #1 BOE

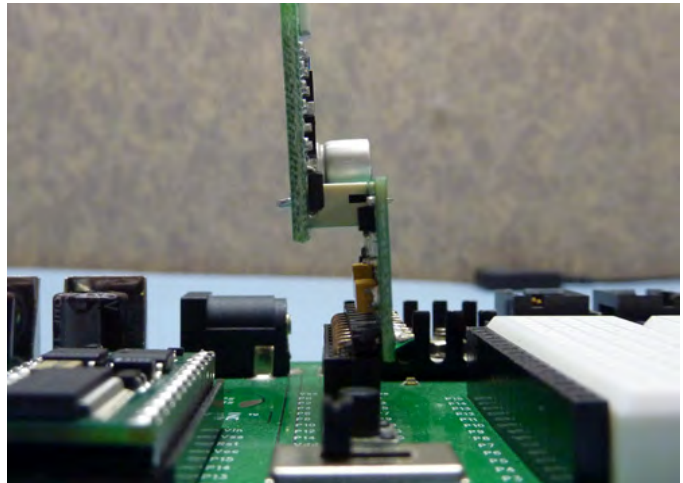
Place one set of rubber feet on the Board of Education using the white circles as the guide for placement.



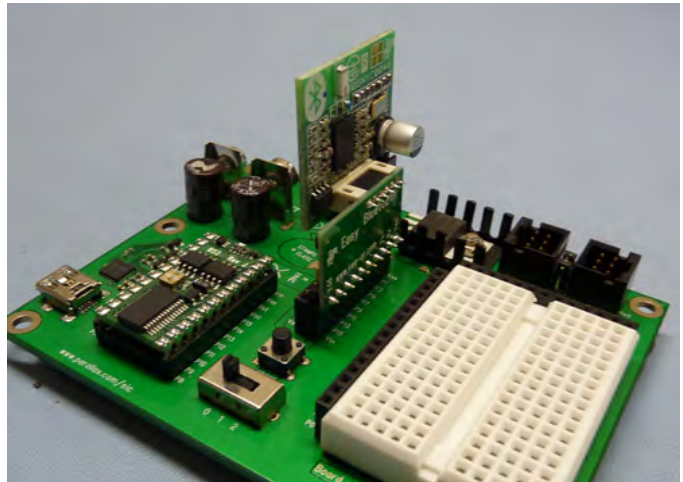
Remove the BS2 from the anti-static foam and place it in the BOE as shown.



Use one of the Easy Bluetooth modules and install the module on the AppMod header. Notice the orientation of the module when installed; the SIP will use the left side of the AppMod header.



Once you have completed the assembly; you should have a BOE with a Easy Bluetooth installed similar to the following.

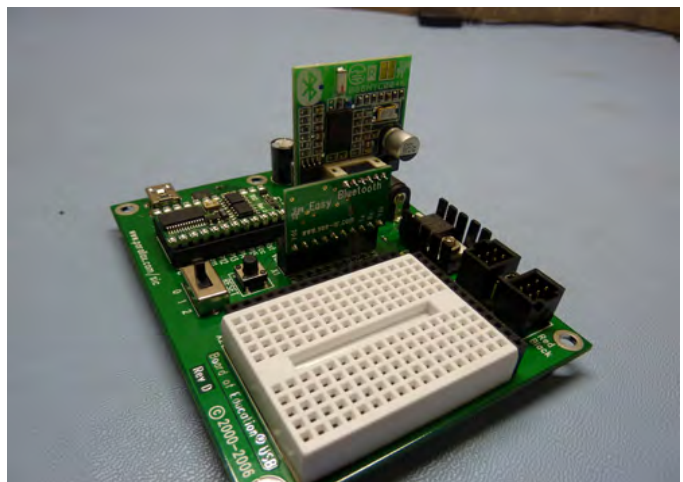


Step 3: Repeat (x 1) and Add Bluetooth Dongle

Repeat step 2 with the other components to assemble the other BOE with Easy Bluetooth setup.

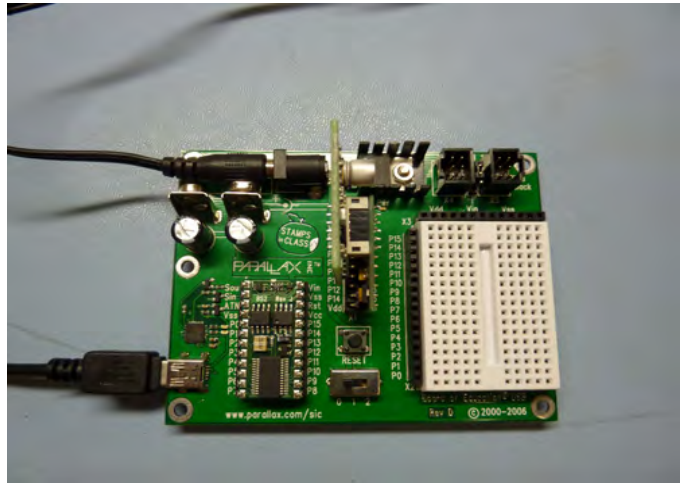
After this step you should have 2 assembled Board of Education setups: BOE1 and BOE2. Make a note or mark each setup with BOE1 and BOE2.

Connect the Bluetooth dongle if you have one or enable the compatible Bluetooth device



Step 4: Connect BOE1

Use one of the USB cables and power supplies to power and connect BOE1 to the PC. Move the power switch to position 1



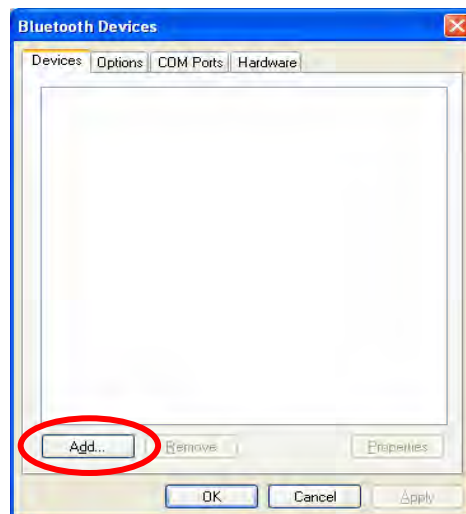
Step 5: Open Control Panel

Open the control panel of the PC, and open *Bluetooth Devices*



Step 6: Establish a Bluetooth COM

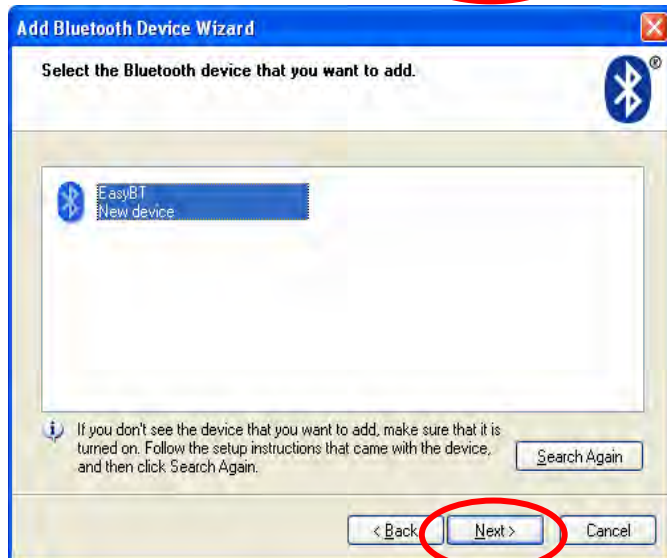
Click Add... to setup the COM ports on the first Bluetooth module on BOE1



Check "My Device is set up and ready to be found" box and click Next.



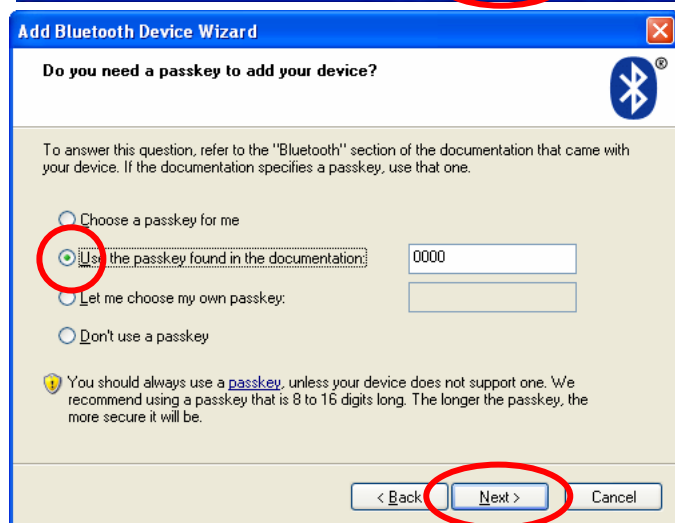
Select the EasyBT module from the device window and click Next.



Check the "Use the passkey found in the documentation".

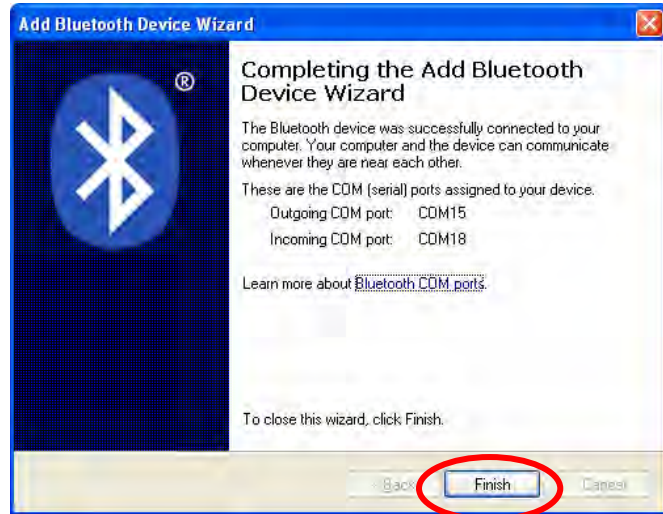
Passkey = 0000
passkey is zero (x4)

Click Next



Click Finish to complete the setup of the BOE1 Easy Bluetooth module.

Note: Make a note or write down the Outgoing and Incoming COM ports that the Bluetooth uses; they can be used later with the Easy Bluetooth for communication with various applications like Hyper Terminal or the Terminal Window in the BASIC Stamp editor.



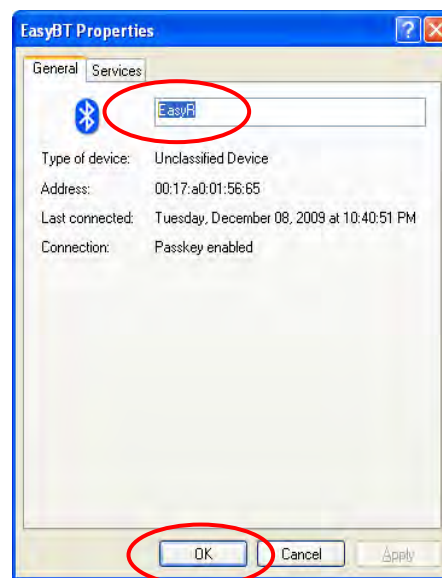
Step 7: Obtain Recipient Modules Device Address

From the Bluetooth Device window, select the Bluetooth module and click Properties. Highlight the name, and change it to EasyR and make note of the address and click OK.

The device address for the module used in this example is:

00:17:a0:56:65

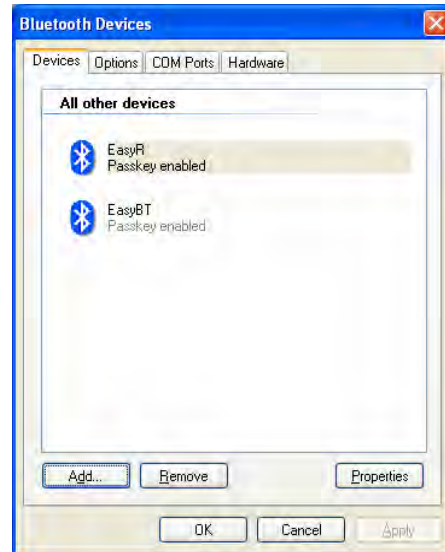
The address is used in the example program for the transmitting Bluetooth module on BOE2.



Step 8: Repeat for BOE2

Power down BOE1 and use steps 4 through 6 to get the Easy Bluetooth on BOE2 configured. Keep note of the Outgoing and Incoming COM ports.

Once both Bluetooth modules are configured, the Bluetooth Device window should look similar to the screen shown.



BOE1 using EasyR

BOE2 using EasyBT

Click OK to close the Bluetooth Device window.

Step 9: Program Editing

Using the address of EasyR (00:17:a0:01:56:65), open the example code EasySEND.BS2 in the BASIC Stamp Editor; then use the address to change the value in the example code.

Place the values for each part of the address in the constants below; whereas the first part of the address will be placed in addy1 and the second in addy2; for example the address A1:B2:C3:D4:E5:F6 would be edited to look like the following:

addy1	CON	\$A1	<--- A1 from A1 :B2:C3:D4:E5:F6 goes here
addy2	CON	\$B2	<--- B2 from A1: B2 :C3:D4:E5:F6 goes here
addy3	CON	\$C3	<--- C3 from A1:B2: C3 :D4:E5:F6 goes here
addy4	CON	\$D4	<--- D4 from A1:B2:C3: D4 :E5:F6 goes here
addy5	CON	\$E5	<--- E5 from A1:B2:C3:D4: E5 :F6 goes here
addy6	CON	\$F6	<--- F6 from A1:B2:C3:D4:E5: F6 goes here

Once the changes have been made with the correct address, download the program to BOE2 and move the power to position 0.

Step 10: Download to BOE1

Connect BOE1 to PC, turn power switch to 1, and open example code EasyR.BS2. Download the example program to the BS2 on BOE1 and turn off the BOE1; close any open DEBUG windows in the editor.

Step 11: Reset BOE1 & BOE2

Leave power off on BOE1 and BOE2 for at least 30 seconds. Disable and unplug the Bluetooth dongle that was used to setup the modules. After delay, turn BOE1 and BOE2 on simultaneously. The connection takes a few moments but there should be visual confirmation that the connection is successful by the green blinking LED on the modules.

Step 12: Open Terminal Window

Open a terminal window in the BASIC Stamp editor (ctrl+d) and select the COM port BOE1 was programmed on; use the default settings in the terminal window.



Resources and Downloads

Source code available from the Parallax web site

Device Information

Theory of Operation

This is a simple demo to illustrate the communication between two Easy Bluetooth modules, on Board of Education, using a BASIC Stamp 2, instead of a PC dongle. The Dongle used in this example is for configuration purposes. Once the communication link has been established one could simply rearrange the information for a number of applications: robotic control, wireless communication, addressable wireless protocols and so on. Enjoy and feel free to share your projects on the Parallax Support Forums. This examples uses Bluetooth to send a variable counter from one to the other; which then displays that value of the counter in the DEBUG terminal; severing the Bluetooth communication will stop the counting in the terminal window.

Precautions

- Do not use the Bluetooth communication signal as the sole safety switch
- Always keep safety in mind when using wireless controlled devices
- Be cautious of the distance needed controlling something wireless; the distance is approx 30m

Source Code

There are two pieces of sample code that go with the AppNote; EasySEND.BS2 and EasyR.BS2. Download EasyBluetoothx2demo.zip file from Parallax web site and load the following examples in the appropriate BOE setups.

BASIC Stamp® 2 Program

Load BOE1: EasyR.bs2

```
' {$STAMP BS2}
' {$PBASIC 2.5}
'
'File: EasyR.bs2
'
'Purpose: Load on the receiving unit in the AppNote Easy Bluetooth to Easy Bluetooth
'Communication.
' The example waits for a value counter to be sent via Bluetooth connection and then
'will display
' the value on the DEBUG screen
'
'Author: Parallax Technical Support

' ' ' ' ' ' ' ' Declarations ' ' ' ' ' ' ' ' ' '

RX          PIN 2
TX          PIN 0

Temp        VAR Byte

#SELECT $STAMP
#CASE BS2, BS2E, BS2PE
  Baud      CON      84
#CASE BS2SX, BS2P
  Baud      CON      240
#CASE BS2PX
  Baud      CON      396
#ENDSELECT

' ' ' ' ' ' ' ' Program ' ' ' ' ' ' ' ' ' '

DO
  GOSUB GETVAL
  GOSUB DISPLAY
  PAUSE 5
LOOP

' ' ' ' ' ' ' ' Subroutines ' ' ' ' ' ' ' ' ' '

GETVAL:
  SERIN RX, Baud, [Temp]                      ' Waits for a BYTE and puts
that value in Temp
```

```
RETURN
```

```
DISPLAY:
```

```
    DEBUG HOME, "Byte = ",DEC Temp,CLREOL,CR ' Displays the decimal value for Temp
RETURN
```

Load BOE2: EasySEND.bs2

```
' {$STAMP BS2}
' {$PBASIC 2.5}
'
'File: EasySEND.bs2
'
'Purpose: Load on the transmitting unit in the AppNote Easy Bluetooth to Easy
Bluetooth Communication.
' The example connects to the remote Bluetooth module via serial connection. An
address is given to the
' remote module that the transmitting unit uses to establish connection. 5 seconds
after the connection
' is attempted then a counter variable is sent and incremented by 1 in a repeat loop
and then will DEBUG
' the counter value.
'
'Author: Parallax Technical Support

' ' ' ' ' ' ' ' Declarations ' ' ' ' ' ' ' ' ' ' ' '

RX          PIN 2
TX          PIN 0

counter    VAR Byte

'Place the values for each part of the address in the constants below; whereas the
first part of the address will be placed in
'addy1 and the second in addy2; for example the address A1:B2:C3:D4:E5:F6 would be
edited to look like the following:

'    addy1    CON    $A1    <--- A1 from A1:B2:C3:D4:E5:F6 goes here
'    addy2    CON    $B2    <--- B2 from A1:B2:C3:D4:E5:F6 goes here
'    addy3    CON    $C3    <--- C3 from A1:B2:C3:D4:E5:F6 goes here
'    addy4    CON    $D4    <--- D4 from A1:B2:C3:D4:E5:F6 goes here
'    addy5    CON    $E5    <--- E5 from A1:B2:C3:D4:E5:F6 goes here
'    addy6    CON    $F6    <--- F6 from A1:B2:C3:D4:E5:F6 goes here

' Place the address for the receiving Easy Bluetooth modules below
' ( the address for the Easy Bluetooth module used in the example is 00:17:A0:01:56:65
' )

    addy1    CON    $00
    addy2    CON    $17
    addy3    CON    $A0
    addy4    CON    $01
    addy5    CON    $56
    addy6    CON    $65

#SELECT $STAMP
#CASE BS2, BS2E, BS2PE
    Baud      CON      84
#CASE BS2SX, BS2P
    Baud      CON      240
```

```

#CASE BS2PX
  Baud      CON      396
#ENDSELECT

' ' ' ' ' ' ' ' Program ' ' ' ' ' ' ' ' ' '

Program_Start:
  DEBUG CLS
  PAUSE 3000

  DEBUG CR,"Establishing SPP connection..."

  SEROUT TX, Baud,
[$02,$52,$0A,$08,$00,$64,$01,addy6,addy5,addy4,addy3,addy2,addy1,$01,$03]
  PAUSE 5000

  DEBUG CR,"Entering Transparent mode...",CR

  'Entering Transparent mode
  SEROUT TX, Baud, [$02,$52,$11,$01,$00,$64,$01,$03]
  PAUSE 3000

  'Transmitting counter variable to remote Bluetooth device and to the DEBUG terminal
DO
  SEROUT TX, Baud, [counter]
  counter = counter + 1
  PAUSE 500
  DEBUG ? counter
LOOP

```